NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

THE USE OF DBASE II AND MICROCOMPUTERS

by

Haralabos Kondylopoulos

December 1985

Thesis Advisor: Linda Rawlinson

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release; distribution is unlimited
2b. DECLASSIFICATION / DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION: Naval Postgraduate School 6b. OFFICE SYMBOL (*If applicable*): 52	7a. NAME OF MONITORING ORGANIZATION: Naval Postgraduate School
6c. ADDRESS (*City, State, and ZIP Code*): Monterey, CA 93943-5100	7b. ADDRESS (*City, State, and ZIP Code*): Monterey, CA 93943-5100
8a. NAME OF FUNDING / SPONSORING ORGANIZATION 8b. OFFICE SYMBOL (*If applicable*)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (*City, State, and ZIP Code*)	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT ACCESSION NO.

11. TITLE (*Include Security Classification*)

THE USE OF DBASE II AND MICROCOMPUTERS

12. PERSONAL AUTHOR(S)

Haralabos P. Kondylopoulos

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (*Year, Month, Day*)	15. PAGE COUNT
Master's Thesis	FROM _____ TO _____	1985 December	150

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (*Continue on reverse if necessary and identify by block number*)
FIELD	GROUP	SUB-GROUP	DBASE II, Microcomputers

19. ABSTRACT (*Continue on reverse if necessary and identify by block number*)

The Hellenic Air Force storage/utilization database management system (DBMS) utilizes mainframe computers. It is a system for large applications, such as big business, and it is used only at the Air Force Headquarters Level. However, the need for an extension of this system, to lower levels, has been recognized for a long time.

This problem can be resolved through the implementation of the DBMS programs developed for this thesis, and the utilization of DBASE II software applied to microcomputers, vice the larger, more expensive, mainframe computers.

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION Unclassified
22a. NAME OF RESPONSIBLE INDIVIDUAL L.C. Rawlinson, Thesis Advisor	22b. TELEPHONE (*Include Area Code*): 408-646-2735 22c. OFFICE SYMBOL: 52Rw

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted. All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE

Approved for public release; distribution is unlimited.

The Use of DBASE II and Microcomputers

by

Haralabos P. Kondylopoulos
Major, Hellenic Air Force

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
December 1985

ABSTRACT

The Hellenic Air Force storage/utilization database management system (DBMS), utilizes mainframe computers. It is a system for large applications, such as big business, and it is used only at the Air Force Headquarters level. However, the need for an extension of this system, to lower levels, has been recognized for a long time.

This problem can be resolved through the implementation of the DBMS programs developed for this thesis, and the utilization of DBASE II software applied to microcomputers, vice the larger, more expensive, mainframe computers.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGEMENTS

I would like to extend my heartfelt thanks to my advisor, CDR L. C. Rawlinson, whose help, support, and patience, were invaluable in the preparation of this thesis.
I would also like to thank my second reader, LCDR P.W. Callahan, who encouraged me by his help and his willingness to answer any of my questions.

I. INTRODUCTION

An amazing amount of progress has been made in the computer field since the primitive computer age of the 1950s. Personal computers, realistic graphics, high-level languages, artificial intelligence, music composition, and many other technological advancements have been made in a period of only 35 years, and new applications are being discovered every day. However, a tremendous management problem has been generated because of the unpredictability of software development time. Several solutions to this problem have been introduced to help the software keep up with the more sophisticated hardware.

In the opinion of the writer, one of the most effective approaches is to let the "same computer" solve the software problem, thus causing increased complexity in the computer. Now the slogan in the field of computer science is: "Fight the complexity with the complexity" [Ref. 1]. That means that we should try to load as much as possible into the computer hardware in order to decrease our working time, since the speed and consequently, the efficiency of the hardware on the modern computer has been increased to the imaginary level.

This concept of recursive construction tries to balance the software development time and cost, by improving and automating the computer process. Starting from a system that controls another less complex unit, we create a bigger system

to control the previous one and another that controls a bigger system to control the previous and another that controls the bigger one and so on. Thus the idea is that we can begin with a minimum of structure, then provide masses of data and wait for the program to solve its own complexities [Ref. 1]. Therefore, an individual works only on the understandable main menus, by using simple instructions without being engaged in the intermediate processes. This progressive aspect of the computer has been enforced in the various fields of computer science, and this principle has recently found greate application in the design of databases.

The idea of recording and maintaining information in an organized manner appeared many years ago, when the value of organized information was realized. The importance of this idea is stressed in the Spinoza expression: "The order and connection of ideas is the same as the order and connection of things" [Ref. 2]. However, the appearance of computers started enforcing this idea with the implementation of applications on the computer.

The use of automation and parallelism theories has also helped the designers to make retrieval of very large databases very easy, and in extremely timely manner. These capabilities of the database have recently been appearing on a variety of modern database machines some of which are briefly described below:

1. VERSO (a relational backend machine): it is being developed at laboratories and its main feature is a hardware filter that can process data at the speed of 2 million characters per second.

2. SABRE (a relational database system for a multimicroprocessor machine): it represents several original ideas in database design such as the clustering of data, multiple views of a database, joint algorithms for relations, query evaluation, and integrity control.

3. Full-Text Information-Retrieval System: can access a database of tens of billions of characters in a timely manner. This satisfactory response time for interactive query formulation and refinement is provided to the system by the finite state automaton architecture mechanism [Ref. 3].

This automation mechanism finds all the combinations of the possible movements from one stage to another, keeps track of the various paths, and selects the optimum one. The magnitude of these machines can be captured by the human mind if we imagine that it can read an entire newspaper in only milliseconds. This tremendous progress in database design has resulted in lower cost, and has provided a strong motivation for working in the database development field, especially on very large databases.

An additionally strong motivation for working in the database field is the wide variety of database applications. These applications include manufacturing with inventory management, the servicing of industries with lists of service capabilities; economic models with production data for allocation and planning, and medical services with patient records, disease histories, problem classification, and treatment effectiveness data [Ref. 2]. Thus, databases are appearing and supporting almost every

science. It might be said that it is the database era in computer application.

An important consideration in the design of the database is the way of storing data, which is used for a broad variety of applications and can be used to make changes to the data quickly and easily. The ability of the database to be applicable in so broad an aspect of applications is based on a common feature that makes database development valuable and general in a programming methodology. This feature is a creative form which is called "structural growth". This "structural growth" should start with a solution on a simplified version of the problem and then repeatedly expand its capabilities up to the desired level. This feature would provide a multiple of "hooks" for the user, ways that the user may insert his/her own code to effect behavior of the system [Ref. 4].

One useful tool that supports this structural development is a working model, a prototype, which has recently become possible only with the development of more powerful procedural based languages. Two other basic tools are the data dictionaries, and strong modularity. In Chapter IV we will examine the capabilities of these tools and we will see how all of these tools provide a strong basis for the relational database management system (DBMS) development.

Among these potential applications, the military requirements for economic, medical, battle planning, personnel

classification and organization, storage organization, and work scheduling information are of primary importance to any defense organization.

Utilizing the advantages of the wide applicability of dBASE II, the objective of this thesis is to provide a generally useful package for military application, and specifically, to meet the requirements of the Hellenic Air Force Storage Organization at the lower levels of utilization, and to enable the Hellenic Air Force Material Staff to more easily handle the queries of "part demand" by using microcomputers.

II. OVERVIEW OF THE MANAGEMENT SYSTEMS

A. WHAT IS A DATABASE MANAGEMENT SYSTEM (DBMS)?

A database management system is a software tool that is used to manage and maintain an organization's database resources. The database management system does no depend on any application program or specific file, but can be used to provide data to several application programs [Ref. 5; p. 335]. A major role of the DBMS is to enable the user to deal with the data in abstract terms rather than as the computer stores the data. The DBMS allows the user to specify what must be done, with little or no involvement in detailed algorithms, or data representation used by the system [Ref. 6; p. 1]. DBMS provide a means of easing some of the problems of managing data in large-scale management information systems. Although the database management system provides facilties and great assistance to the user by reducing much of his/her potential errors, it is one of the most complex variations of software in existence.

B. INTERACTION OF THE PROGRAMMER/USER WITH THE SYSTEM

The description of the database is written in a specialized language or Data Definition Language (DDL), and is compiled into tables that exist in the system. Programs that are permanently stored in the DBMS which are used to manipulate the data are called application programs. A collection of

routines that is used to process queries is called a database manager. A database manager translates the query into terms that the file manager understands. The database manager prevents the duplication of data by storing everything in a single database, it reduces the program development time by using many routines, and it improves the reliability of the system by automatic integration of information and the relationship of information [Ref. 7: p. 14].

The file manager may be provided by the underlying operating system, or may be a specialized file system that contains all the information about the storage of the data. The database manager with the assistance of the file manager, can provide rapid access to, and manipulation of the database. The database manager usually performs other tasks, such as the following:

1. Security. Gives the ability to the user to protect the access of its data from unauthorized persons.

2. Integrity. Enables the DBMS to check the data whenever a user uses a command to insert, delete, or change some data.

3. Synchronization. Provides protection to the DBMS against inconsistencies that are caused when two or more approximately simultaneous operations occur on a data item [Ref. 6: p. 4].

In figure 2.1 we can see the components that comprise the whole system.

C. LEVELS OF ABSTRACTION IN A DBMS

A fairly standard viewpoint regarding levels of abstraction is the three levels of presentation (views, conceptual

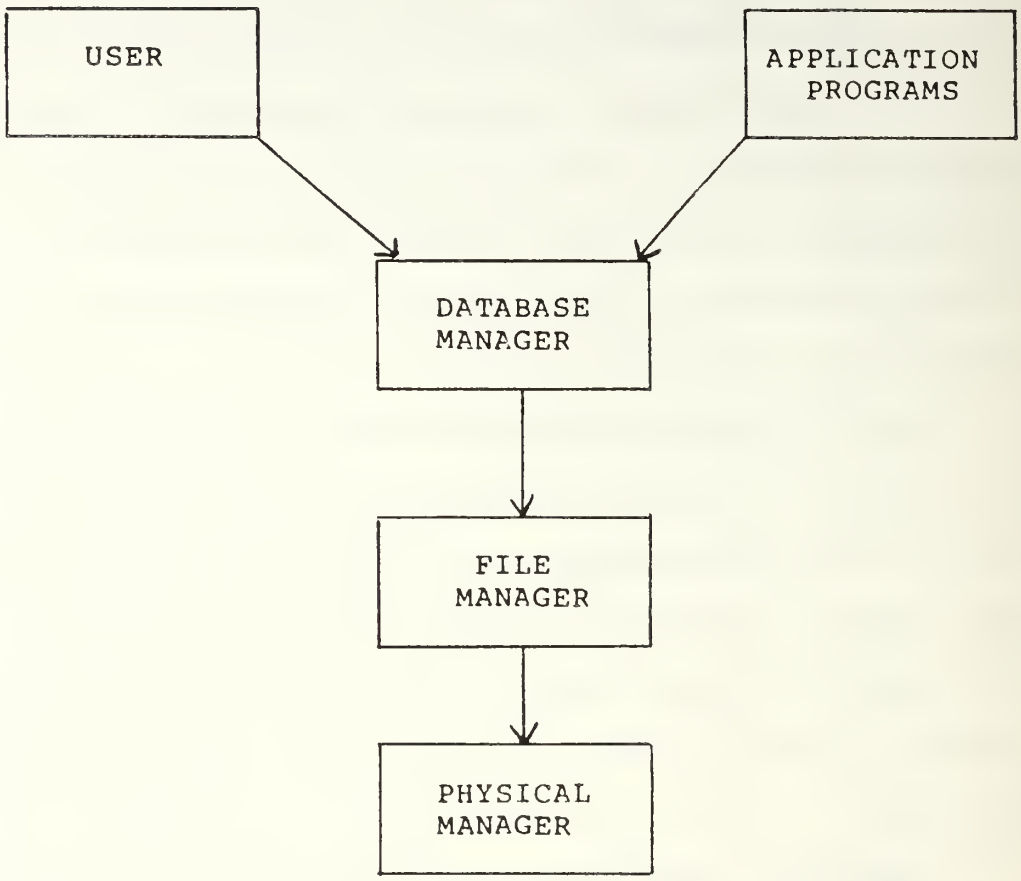

Figure 2.1. Schematic Diagram of a Database System

database, and physical database), which are shown in figure 2.2.

1. The views are an abstract model of a part of the conceptual database. A facility for declaring views, called a subscheme data definition language, and a facility for expressing queries and operations on the views, are called a

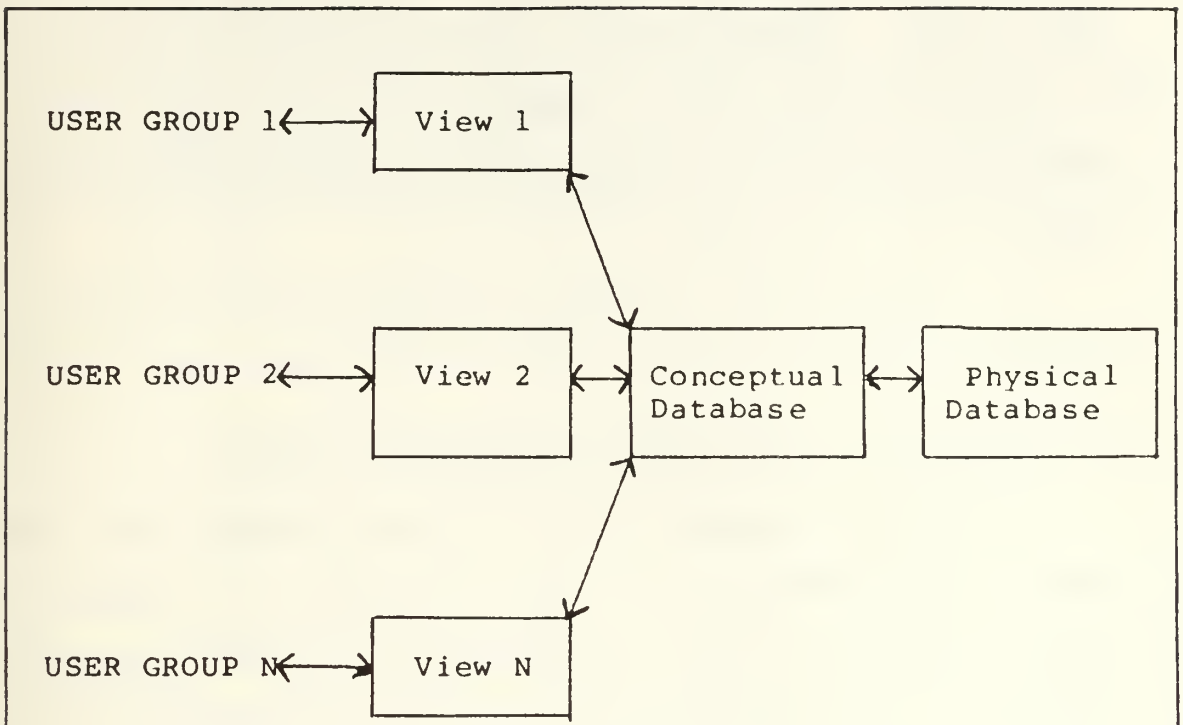

Figure 2.2. Levels of Abstraction in a Database System

subscheme data definition language, and a facility for expressing queries and operations on the views, are called a subscheme data manipulation language. An example of the use of the different views could be set, an airline with a computerized reservation service system, that is concerned only about the flights and passengers. However, another system (view) would be concerned about personnel files or assignment of mechanics to work on specific airplanes [Ref. 6: p. 6].

2. The conceptual database is just an illustration of the real world. A DBMS provides a data definition language (DDL) to specify the conceptual database. DDL is a high-level language that is used to describe the conceptual database in regards to a data model.

3. The physical database is stored permanently on secondary storage devices, such as disks and tapes.

D. BENEFITS OF THE DATABASE MANAGEMENT SYSTEMS

A database management system can be a complex and expensive tool for an organization. The installation of a database management system increases substantially the entire data processing cost of an organization, since the initial cost may range from $10,000 to over $200,000. For that reason an organization must compare the overall costs of a DBMS with the benefits to be gained from its use. The advantages of DBMS come from:

1. Improved control. The DBMS improves the control of data in numerous ways. Data can be organized and structured so that it is suitable for many application systems. In addition, control over input and use of the data is improved. Control is also improved for privacy and security of data by using the built-in procedures of the DBMS.

2. Evolvability. The ability of DBMS to be adjusted to meet changes caused by either usage requirements or technological improvements.

3. Service. Service may be improved at a lower total cost to an organization because of the more efficient use of storage space and substantial reductions in program maintenance costs. The reduction of the program maintenance cost is importance since typically more than half of the expenses is absorbed by the software systems.

4. User Benefits. In addition to the previously discussed benefits, we can add some others that go directly to

the user. Users, by using the different skills from the different levels of languages, can interrogate the database to "fine-tune" requests. Thus, they receive more appropriate information, with less of the irrelevant information often associated with some types of generated reports. Similarly, through direct interrogation, users can find answers to those un-anticipated inquireis that play an important role in rapidly changing organizational environments [Ref. 5: p. 352].

III. BASIC MODELS OF THE DBMS

There are three basic models of the database management system. There are the network, the hierarchical and the relational models (figure 3.1).

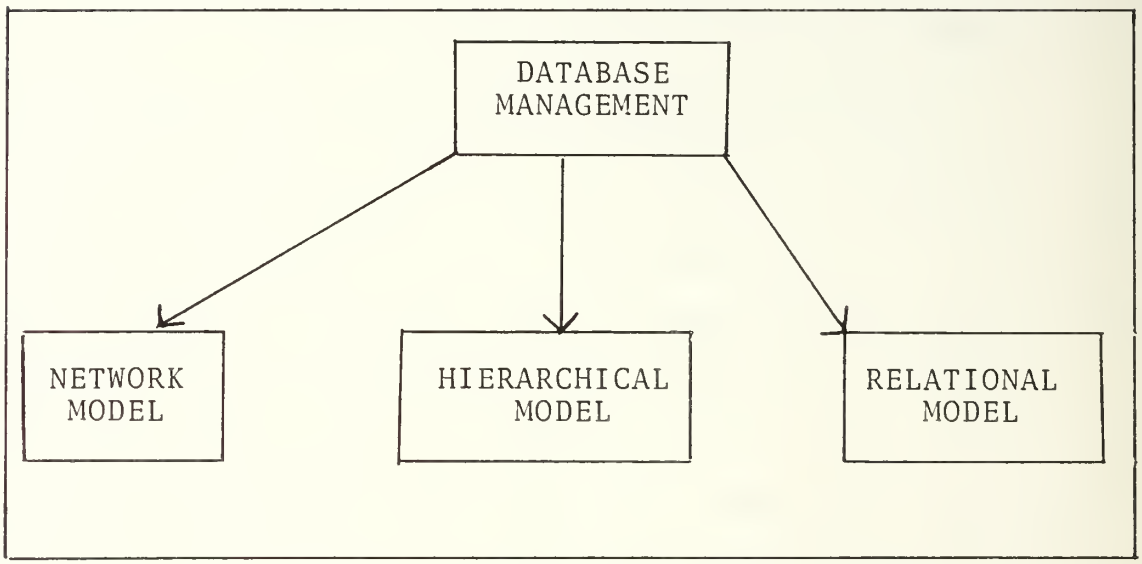

Figure 3.1. The Three Database Models

A. Hierarchical

A hierarchical model is a special case of the network model. A hierarchy structure can use ordering conventions for the segments. The characteristic of this model is that each descendand (child), has to have, at most, one link with the other file. In addition, this link must proceed only from ascendand to descendant. Thus the basic operation on a hierarchical database is a tree walk; that is, given a node (which represents a record of the database) of the database

tree, we search all its descendants of this node. The figure 3.2 shows the hierarchical model of the same organization depicted in the network model.

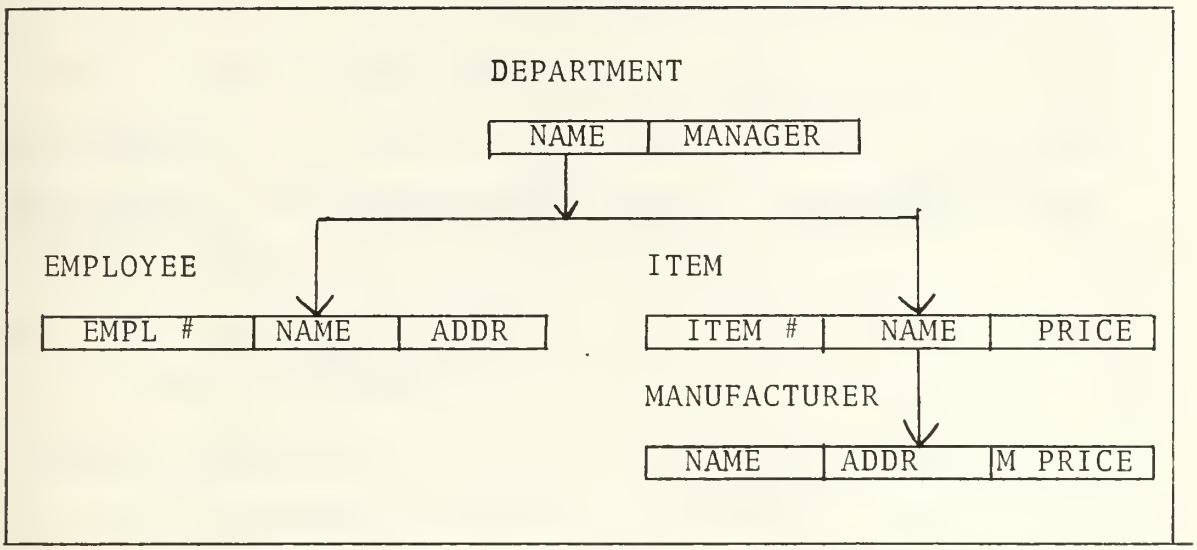

Figure 3.2. A Hierarchical Model

B. NETWORK

A network is used when applied structures are more complex than hierarchies, a special case of the network model. The characteristic of the network is that its links are bi-directional, allowing us to travel either from many to one or from one to many files. The figure 3.3 shows the schematic representation of one organization by using the network system.

C. RELATIONAL

The relational model is quite different in its organizational structure. Information is stored in two-dimensional files, and the manipulation of its queries uses the relational calculus. It uses indexed tables for fast search and

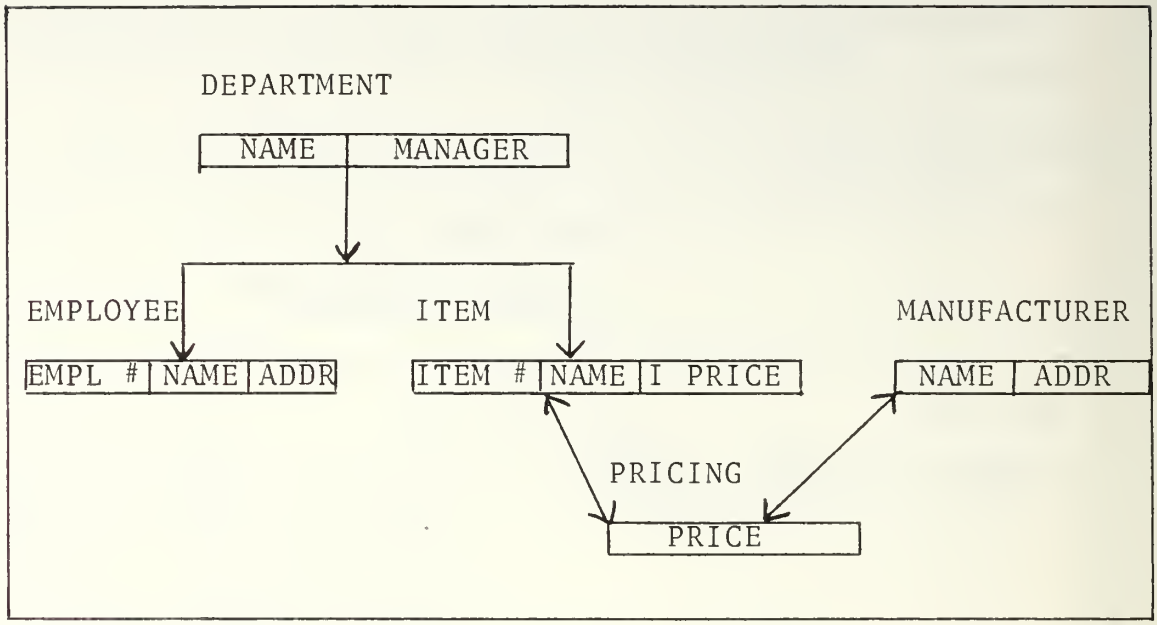

Figure 3.3. A Network Model

retrieval. The uniquely identified attribute of an indexed file is called a key, and is represented by underlining the attribute showed in figure 3.4.

DEPARTMENT (DNAME, MANAGER)

EMPLOYEE (EMP #, NAME, I PRICE)

ITEM (ITEM #, NAME, I PRICE

MANUFACTURER (MNAME, ADDR)

Figure 3.4. A Relational Model

D. COMPARISON OF THE MODELS

The relational model is easier to use than the other two models, because it provides only one concept, the relationship, that the programmer must know very well. Further, the

network model requires our understanding of both record types and links, and their relationships. The implementation of "many to many" (linking from many records of one file to many records of another file) for more than two files provides serious complexity. Similarly, the hierarchical model has the same problems as the network has, because it requires understanding the use of the pointers.

From the aspect of efficient implementation, the network and hierarchical models clearly win. This happens because of the difficulty of the relational model to implement "many to many" relationships. Here we should add an intersection file (with the fields including at least the keys of the two files), that causes problems and creates complexity on the system.

The last criterion of the model's evaluation is the language that supports each of them. From this aspect we note that the relational DBMS requires support by high-level languages, while DBMS based on other models have a tendency to use languages of a lower-level. The high-level language provides a single "data type" for the relationship. In contrast, lower-level languages enable the programmer to use a variety of data types. This variation of the data types used in the network and hierarchical models provide efficient implementation, but require skilled users.

IV. TOOLS THAT SUPPORT A DATABASE

The increasing productivity of systems development, the shortening of the response time of the computer, and the increasing complexity of computer systems, require effective tools that will be capable of processing information. This chapter covers advanced tools currently available to meet these challenges. In the opinion of the writer, such tools as prototyping, data dictionaries, and modularity, are the appropriate choices.

A. PROTOTYPE

1. What is Prototyping?

Prototyping has proven to be a valuable technique throughout the engineering sciences. Its name comes from the Greek words "protos=first" and "type=model" [Ref. 2: p. 26]. It is not a new concept. Software systems are being constructed today in much the same way that they have been for the past two decades. In some instances, scientists and engineers with sophisticated support environments have used prototyping as an alternative approach. They realized that "pilot systems", are necessary and useful tools which reduce the inherent technical risks created with developing new software. Prototyping is a hand-crafted version of a final production model [Ref. 8: p. 14]. "Prototyping" is also defined as: "a perfect example of a particular type" and as

"an original or model after which any thing is formed" [Ref. 9: p. 13]. It is a technique which offers the probability of delivering a clear, correct, and validated requirement statement of the user's need.

2. Prototype Life Cycle

The development steps that a system should follow, when using a traditional management information system approach, are shown in figure 4.1, and the development steps of a system using the prototyping techniques are shown in figure 4.2.

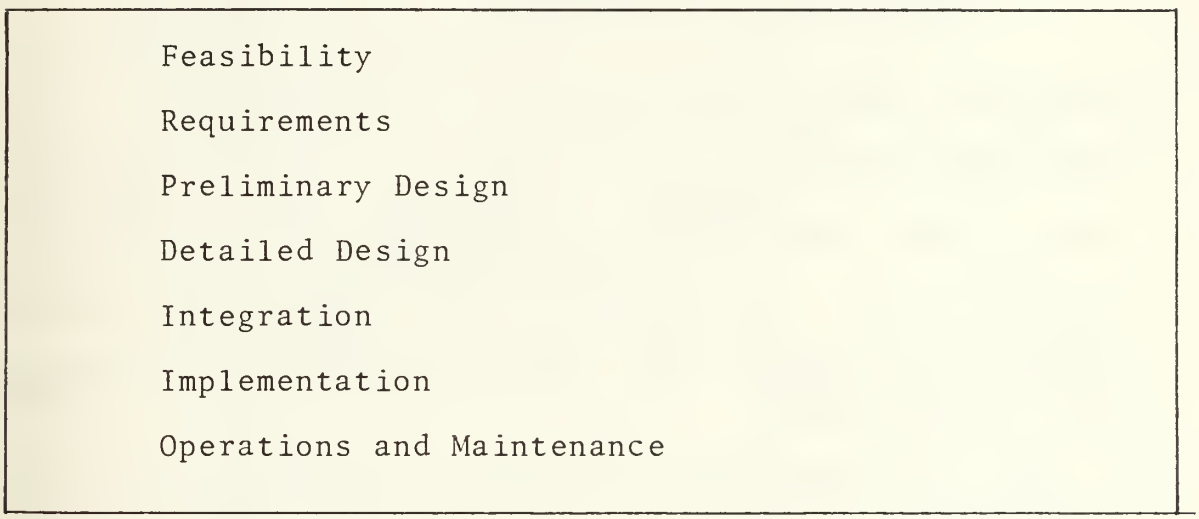

Figure 4.1. Traditional Life Cycle Technique

The figure show that the only difference from the traditional technique is in the requirement steps, which appears in the prototype technique [Ref. 10: p. 163]. On the substeps of the prototyping development, we can see that the key-mechanism of the structure is the iteration process. The iteration process gives us a high-degree of confidence in the design quality and completeness of the design [Ref. 11: p. 25].

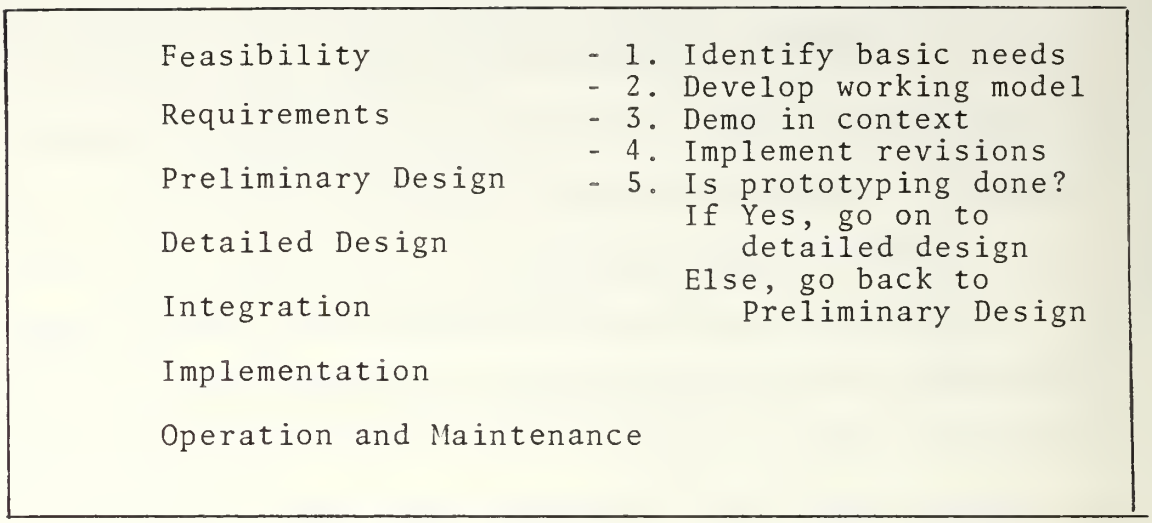

Figure 4.2. Prototype Life Cycle Technique

3. Advantages of a Prototype

The main benefit of prototyping is its ability to bridge the communication barriers in the development process. This feature enables the prototyping to create a dialogue between human and machine, allowing the user to make minor changes quickly. In addition to this benefit, the prototyping approach has numerous other advantages during the development of a system, such as:

- Prototyping technique is usually faster than traditional methods.
- It permits assessment of the impact of the system on the entire user environment.
- It permits early testing of the human/machine interface.
- It provides a medium for validating requirements.
- It requires fewer programmers.
- It reduces the life cycle costs by reducing the maintenance cost.
- It is developed with correctness in mind [Ref. 2: p. 26].

4. Prototyping Limitations

In spite of the numerous advantages of prototyping, it does have limitations. The major limitations are:

- Prototypes are not developed with efficiency in mind [Ref. 2: p. 26]. (This is caused by the decrease in software development time and the use of fewer programmers).
- Prototyping does not necessarily result in shorter development time.
- It is possible for a programmer to leave out some major part of a system during the development phase.
- Because of extensive use of the iteration process, time estimating procedures are difficult.
- Prototyping is not appropriate for all types of applications.

Although these limitations seem to degrade the value of prototyping, it is clear that most of these problems can be solved as more experience is gained.

5. Prototyping Applicability

The greatest potential advantages of prototyping appear on large systems development. Algorithm based problems do not create an appropriate environment for fast iteration. On the other hand, on-line transaction processing based applications work perfectly by using the prototyping technique. Prototyping can also be used as a basis for bottom-up integration. It can do an evaluation of update and retrieval efficiency, and can be used to simplify the implementation and acceptance of a new system.

Prototyping can also be used to improve the physical database design. In order to understand this improvement,

we can think of the relational database as a "pilot model" which is the structure of the database file, and then the iteration feature of the prototyping will be applied sequentially on all records of the same file.

However, in addition to the assistance that is provided by prototyping in the database software development design, the data dictionary tool should also be available for use.

B. DATA DICTIONARIES

A data dictionary is used for the definition and the recording of the data elements, and the handling of the logic representations such as decision tables, and structural English. It is a document that provides details of each data file that is described [Ref. 12: p. 186].

A typical data dictionary entry is presented in figure 4.3 [Ref. 13: p. 126]. It should include the name of the data item, where it is used, its purpose, where it is derived from, its subordinate systems, and any other appropriate notes.

We can see that the relations are composed from fields of the data records. These fields are called domains of the relation. For example, the record "ORDER" has seven fields as in figure 4.3. ORDER ("customer-identity, order-date, catalog-number, item-name, unit-price, quantity, total-cost).

```
DATA FLOW: ORDER              DATA STORE: ORDER-DATE

COMPOSITION:                  COMPOSITION

    CUSTOMER-IDENTITY             TIME
    ORDER-DATE                    DAY
    ITEM-ORDERED                  MONTH
        CATALOG-NUMBER            YEAR
        ITEM-NAME
        UNIT PRICE
        QUANTITY
        TOTAL-COST
```

Figure 4.3. Example of Data Dictionary

The value of dictionaries to the maintenance staff of a system are obvious. Much of the information conventionally gathered by cross reference programs can be entered into the dictionary, and when a program object is to be changed, reference to the dictionary should be made to ensure the changes are accepted by the system [Ref. 12: p. 187].

Since the data dictionary can define attributes of the data, and relationship among them, it can be utilized as a source of information that is needed by prototyping.

A data dictionary may also contain the output of prototyping in an organized database, and generally should be the main factor to integrate all the conceptual output that is produced by the tools that are being used [Ref. 14: p. 250].

C. MODULARITY

Modulus always refers to parts that can be put together to make a complete system. We can think of the modulus as a

work assignment given to a programmer or group of programmers in order to construct a complete system by putting their assignments together. The use of modulus and subsystems provide a powerful abstraction mechanism, since the parts of the whole system can be compiled separately, they can use their libraries, and they can link the library units into the object code. Systems with strong modularity consist of sell-defined manageable units, and with well-defined interfaces among the units. For well designed systems, the modularization criteria should have the following characteristics:

1. Modules which contain instructions, processing logic, and data structures
2. Modules which can be separately compiled ans stored
3. Modules which can be included in a program
4. Modules which can utilize other modules [Ref. 13: p. 147].

All of these characteristics make the modularity powerful and very useful throughout the computer sciences, and provides effective solutions, especially in the large and complex programs.

In the relational database, modularity can be used to integrate the developed structure of the system in order to be more understandable and easier to handle by the user.

Examples of modules include procedures, subroutines, and functions. Modularization permits the designer to separate a system into functional units, to establish the hierarchical

feature of the ordering, to implement data abstractions, and to develop software by using subsystems [Ref. 13: p. 147].

V. OVERVIEW OF MICROCOMPUTERS

A. INTRODUCTION

In general, by the term "microcomputer" we mean a stored computer program comprising memory and input/output circuits together with a microprocessor (CPU), residing on one or more chips. Microcomputers are a recent technology and are accepted by many people because they provide several advantages as compared to the larger computers. First of all, microcomputers are not as expensive as the mainframe, and can be very helpful for many people for personal use. Second, microcomputers are powerful, reliable, and can be used for a wider range of specific applications. Third, they are acceptable to any environment and can replace the older computers, which require additional funding for maintenance personnel and special facilities (air conditioned rooms and large spaces). However, the limited access speed and the limited storage capabilities are the major disadvantages of microcomputers [Ref. 15: p. 11].

B. PROBLEMS AND CHARACTERISTICS OF THE MICROCOMPUTER-BASED DBMS

The use of a database management system (DBMS) in the microcomputer environment is a relatively new orientation. Recently many DBMS packages have been tailored for the microcomputer. They have become very attractive to the user because of the high cost of the larger computers and the

requirement of less memory storage for the smaller systems. In spite of the fact that the new packages were developed independently of the mainframe DBMS, they still have similar characteristics and problems, such as:

1. Concurrency control. The special problems which take place when a database is accessed by many users. For example, when two users simultanteously lock and access two records, and then they both attempt to lock the second one which is locked. This problem (deadlock) disappears when the record is unlocked. DBASE II does not use any mechanism to enforce concurrency.

2. Transportability. This issue is focussed on two levels. Portability of application programs and DBMS in high-level language and portability of DBMS machine. DBASE II is offered on CP/M and compatible systems. Since it is interpreted at run time, only the interpreter is modified to run on other systems. The language of DBASE II remains the same.

3. Integrity and consistancy. The ability of the DBMS to determine if the entered data is appropriate and valid for further processing. DBASE II has rather limited consistancy checks and it does not provide a range limit on all insertions.

4. Security. This important feature should concentrate on several things. The prevention of unauthorized access and modification to data in secondary storage; the prevention of unauthorized access and modification to data when the DBMS is in use; and a provision of selective security to control access to the individual fields of a record. DBASE II does not provide any explicit security. We could use a password protection by creating a specific command file, but this protrection could be by-passed by reading directly to the command files.

5. Crash protection and recovery. The ability of the system to recover in case of a hardware failure. Typical problems which take place are disk damage, temporary power failure, and transients in the power supply which cause data in memory to be mixed, etc. DBASE II has no special functions to protect the system from damage caused by the hardware. The only protective operations are the echoing of the operations and transactions to the printer or a disk. Consequently a backup copy is the best way of recovering from such problems.

C. DBASE II CONCEPTS

DBASE II is a relational DBMS. However, since the relationship of the different files of the same application is not stored in the system, we can not say strictly that it is a true DBMS. The main advantage of DBASE II is that it provides simplicity in application programming. This means, it can be easily used by programmers with a minimum of experience. DBASE II can be applied to the financial, manufacturing, home, and business fields. Some typical applications include:

* General ledger, accounts receivable, accounts payable, payroll, job costing, and legal office accounting.

* Inventory control, purchase order systems, project management, time scheduling, materials requirement planning, invoicing systems, and order entry systems

* Stock portfolio management, games, simulations, nutritional analysis, tape and record collection management, coupon exchange management, recipe file control, home budget analysis, and tax computation

* Real estate management, mailing list management, ticket ordering system, time billing, and personalized mass mailings.

Some of the features of DBASE II are:

* A large degree of program and data independence. Data structures can be altered without the necessity of many program changes

* Data can easily be added, edited, deleted, sorted, indexed, or reported using a minimum of programming.

* Reports can easily be created from data in a database using mathematical operations like multiplication and division. Subtotals and totals can be easily generated.

* Professional screen formats can be created for data entry. You design the entire screen, and data entry only requires the operator to "fill in the blanks."

* The internal language is extemely powerful with single commands for indexing, sorting, and reporting. Often programs can be developed at five or ten times the speed of developing equivalent BASIC programs.

* The internal language is a structured language, and its programs are far easier to write and update than unstructured programs [Ref. 7: p. 12].

VI. APPLICATION OF DATABASE DEVELOPMENT

This chapter describes the steps taken to develop the Hellenic Air Force Storage/Utilization database management system, and provides the course application programs to support the needs of the user. Example reports are also included to demonstrate how this database can be used to provide better communications and user satisfaction.

The phases that comprise this development are: the analysis phase, design phase, and the implementation phase.

A. ANALYSIS PHASE

1. Introduction

The analysis phase of software development involves requirement specifications that state the "what" of the software without implying the "how." The "how" specification concerns software design and will be examined in the design phase section.

In reference to this thesis, the demands of the Hellenic Air Force Storage System (HAFSS) must be specified and the objective requirements for the system that is being developed must be analyzed.

2. System Objectives

The wide variety of models of same-purpose weapons, the different types of airplanes, the different measuring systems, the different kinds of airplane parts, and the

large number of users that manipulate these parts, has recently created a major problem for the Hellenic Air Force Storage/Utilization System. Actually the solution of this problem is becoming more complex as the demands for more effective and more complex weapons systems emerge.

As a partial solution to this problem, computerization was introduced on an Air Force level basis in recent years. It is the intention of this thesis to suggest an expansion of this system, down to the Air-base level, in an attempt to increase efficiency in the handling of airplane parts.

As a facet of this expansion to the base level utilization, microcomputers are strongly suggested by this thesis for reasons which were described in Chapter V. Their utilization on sequentially lower levels of the organization will revolutionize the current process of stores accounting, and provide the necessary measures of economy of manpower, ensuring that sufficient stocks are maintained in a timely fashion. In this utilization, (and due to the limited storage capacity of microcomputers), care must be taken to store only that data which is pertinent to the tasks required. The kind of output that must be retrieved from the system vary, and depend on several factors. To meet the demands of the Hellenic Air Force Storage/Utilization System, the application software developed from this study, is based on the appropriate inputs and is directed by the following necessary outputs.

INPUT	OUTPUT
PART NUMBER	WHERE CAN WE FIND A SPECIFIC PART?
PART-NUMBER, DATE	HOW MANY PARTS WERE ORDERED IN A SPECIFIC TIME PERIOD?
USER-NAME, DATE	WHO ORDERED THE PART AND WHEN WAS IT ORDERED?
PART-NUMBER, DATE	WHICH PARTS DO WE NEED TO REORDER?
SUPPLIER-NAME, COST PART-NUMBER	WHICH SUPPLIER IS PREFERRED FOR BUYING A SPECIFIC PART?
PART-NUMBER, QUANTITY	WHAT IS THE REQUIRED QUANTITY OF A SPECIFIC PART?
INVOICE-NUMBER, PART-NUMBER	BY WHICH INVOICE NUMBER WAS A SPECIFIC PART TAKEN?
PART-NUMBER, DATE, COST	WHAT WAS THE COST FOR A SPECIFIC PART IN A SPECIFIC PERIOD?

The question remains as to "how" to implement these objectives. The "how" is answered in the design phase of the application's development.

B. DESIGN PHASE

According to Webster, the process of design involves "conceiving and planning out in the mind" and "making a drawing, pattern, or sketch of" [Ref. 13: p. 137]. In software design, we have two separate kinds of activities. The external design is concerned with refining the software requirements and establishing a bird's eye view of the structure of the system and the internal design is concerned with the internal structure of the system and the processing details. Design is

the connection between requirements and an implementation that satisfies those requirements, and is the "how" of the software products.

The database system of this thesis organizes the data into files, and the data stream is accomplished by the command programs of the system. The use of menus and submenus is an ideal way to establish the top-down design were attention is first focused on global aspects of the overall system, and then is decomposed into subsystems where more consideration is given to it.

Figure 6.1 illustrates the organization of the main menu and submenus of the system (by capital letters), and their programs (lower case letters).

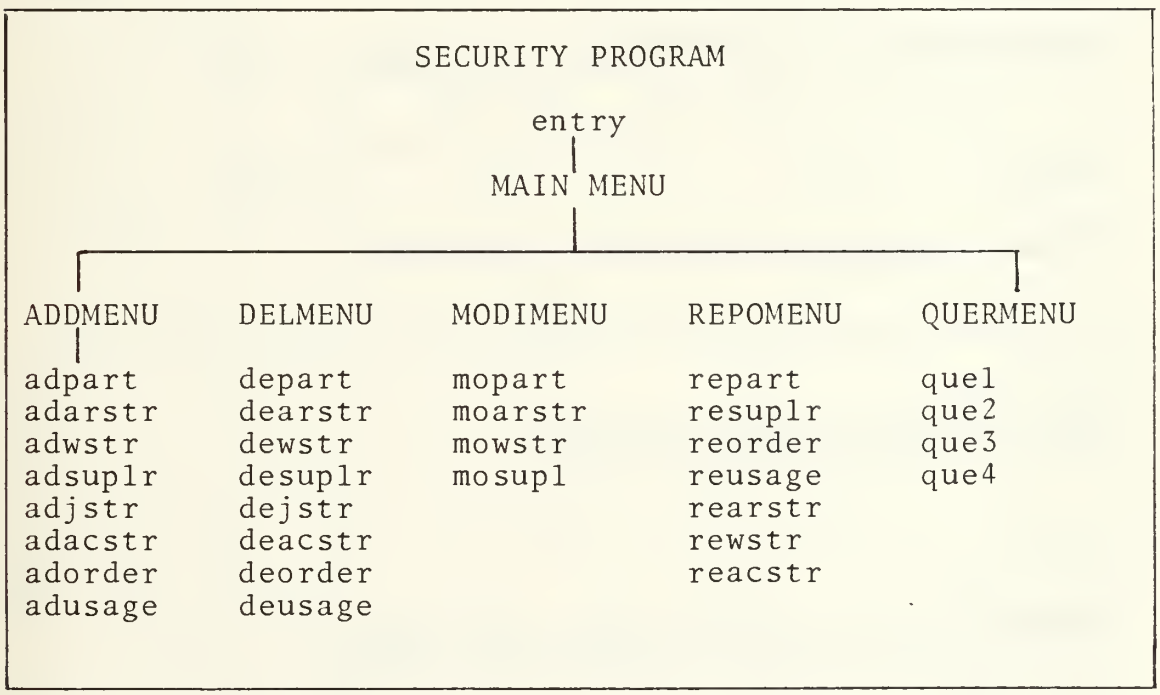

Figure 6.1. Organization of the Database Programs

FIELDS OF THE DATA FILES OF THE AIRPLANE PARTS SYSTEM

PARTS

PNO	DESC	ARPLT	SUPLRNA	ARSTRNA	WSTRNA	JUNA	ACSTRNA	QOH	COST

SUPPLIERS

SUPLRNA	SUPLRADR	ZIP CO	COUNTRY	ARPLT	DELIV TIME	PRODUC

ORDERS

ORDNO	DATE	PNO	DESC	ORDNA	SUPLRNA	QNT	COST
-----	----	---					

USAGE

INVNO	DATE	PNO	DESC	UNAME	UADDR	QNT	COST
-----	----						

ARSTORES

ARSTRNA	PNO	DESC	ASLFNO	ARPLT	QNT	COST
-------	---					

WSTORES

WSTRNA	PNO	DESC	WSLFNO	ARPLT	QNT	COST
------	---					

JUNSTORES

JUNA	PNO	DESC	JSLFNO	QOH	COST
----	---				

ACSTORES

ACSTRNA	PNO	DESC	ACSLFNO	ARPLT	QOH	COST
-------	---					

Figure 6.2. Organization of the Files

The figure 6.2 shows the structure of the files of the system that are constructed according to the specific demands of this system. Also, a hierarchical style representation of the files is illustrated in the figure 6.3. Meaning of the names in figure 6.2 follows:

PNO = Part Number	DESC = Description
SUPLRNA = Supplier Name	ARSTRNA = Stores of Airpl Parts
WSTRNA = Weapon Stores	JUNA = Junk Stores
ACSTRNA = Accessories Stores	QOH = Quantity On Hand
COST = Cost	SUPLRADR = Supplier Address
ZIP CO = Zip Code	COUNTRY = Country
DELIV TIME = Delivery Time	ORDNO = Number of the Order
DATE = Date	ORDNA = Name of the Orderer
INVNO = Number of the Invoice	UNAME = Name of the User
ASLFNO = Airpl Store Shelf Number	WSLFNO = Weapon Store Shelf Number
JSLFNO = Junk Shelf Number	ACSLFNO = Accessories Shelf Number

The requirements of the analysis phase that have been discussed in the preceeding section, are satisfied by using the above files, and by using the information stored in the fields of those files. All the necessary operations will be performed under the program's control. For example, in order to satisfy the first question of the analysis phase, "where can we find a specific part" (with part # 222222), we have to use

the indexed "parts" file with the key "PNO", and go through it until we find the record with the number 222222. Furthermore we can obtain the information on where this part is stored, including its location, the shelf number on which this part has been stored, and the quantity-on-hand.

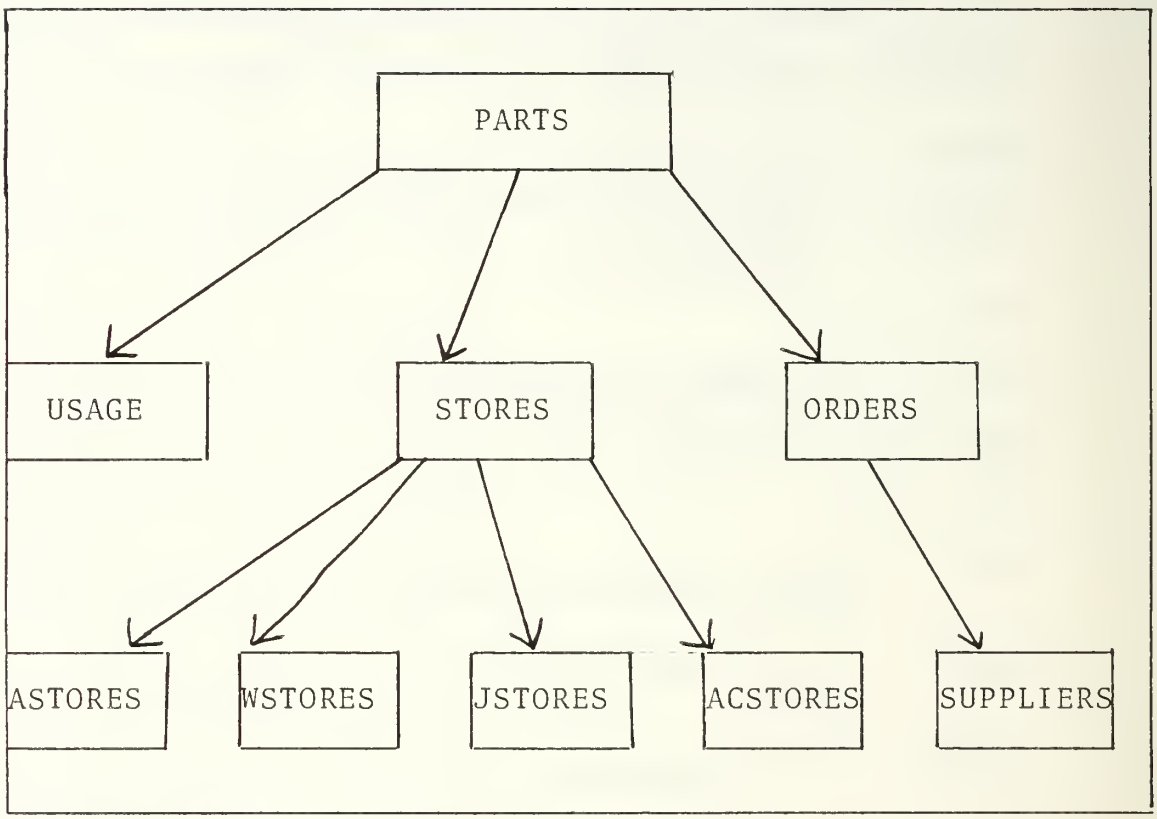

Figure 6.3. File Organization in Hierarchical View

C. IMPLEMENTATION PHASE

This section describes the implementation of the database system of this thesis, by using the relational DBMS, DBASE II. Generally, in the operational environment, the DBMS provides several functions to carry out the implementation of the application programs. However, although the

principal function of the DBMS is to store, retrieve, and modify data, the DBMS provides other important functions as listed below:

1. Store, retrieve, and update data
2. Provide integrity services to enforce database constrains
3. Provide a user-accessible catalog of data description
4. Control concurrent processing
5. Support logical transactions
6. Recover from failure
7. Provide security facilities
8. Interface with communications control programs
9. Provide utility services [Ref. 18: p. 408].

Ideally, for a product to be a relational DBMS, it must be a full-capability DBMS, plus it must process data as relations. In practice, no relational DBMS product concurrently provides all of the nine functions.

The basic characteristics that a DBMS must have to be considered as a relational model, are tables that define data and are processed by SELECT, PROJECT, and unrestricted JOIN operations. DBASE II substitutes the principle relational DBMS commands with the following ones:

"FOR" for SELECT	"JOIN" for JOIN
"GET" for RETRIEVE	"COPY" for PROJECTION
"UPDATE" for UPDATE	"FIND" for FIND
"STORE" for STORE	"CREATE" for CREATE

Appendix A of this thesis contains the listings of the application programs that has been developed by handling these DBASE II commands.

The organization of these application programs include: addition, deletion, modification, reports, and queries. Furthermore, each of these operations is applied to each of the database files of the Airplane's Part System.

1. Addition Operations

The addition operations are applied to all of the files of the system, and their purpose is to add new data to the appropriate file. The addition is accomplished interactively by program control through the use of menus.

The addition process starts by asking the user to enter the specific data (part, user of the part, ordered part, etc), by filling some blanks on the screen with characters or numbers. Then the new data is stored into computer memory, displayed on the screen for verification by the user, and then copied into an appended blank record. In case of the part addition, entries are added to the part (master) file, plus the appropriate file. For example, a weapon part is stored into weapons data.

2. Deletion Operations

Deletion operations like addition operations are applied to all the files of the system, and are very fast operations. These operations enable the user to delete data from the appropriate file interactively. The process starts

by asking the user to fill some blanks of the screen, in order to define the specific data that is going to be deleted. The program then, takes this input, uses the indexed file, finds the data of the specific record, displays all the record data to the screen for an additional check by the user, and ask him/her again if he/she desires to delete this record. In the case of deletion operations from the part file, the specific part is deleted from the part (master) file, and from the appropriate file.

3. Modification Operations

The purpose of the modification operations is to allow the user to change the data of a specific record. This process starts by asking the user to input through the screen the specific data that he/she is interested in modifying. Then this entry by the use of the indexed file locates the appropriate record, and displays its data on the screen enabling the user to modify the desired data. The new entries change the appropriate record on the part (master) file, and the affected file that keeps the specific data.

4. Report Operations

The report operations enable the system to report the current parts of the airplane stores. The menu driven capability helps the user to get a report for any store he/she desires. In addition to store reports, the reports for the airplane stores include a submenu in order to provide to the user partial report for a specific kind of parts. For

example, in the airplane-part store, the user can take (printed or displayed) reports of the part for the Fighter F-4.

5. Query Operations

The purpose of the query operations is to allow the user to ask the most important questions that could retrieve from the system. Since the number of the queries can be too large, a good cooperation between the designer and the user is a must in order to design the system to operate only on the appropriate aspect according to the needs.

VII. CONCLUSION

The increased demands of the Hellenic Air Force Storage/ Utilization Airplane Parts, created the need of decentralication of the database management system which is based on the use of mainframe computers, and performed at the Air Force Headquarters Level.

This thesis developes a database application package which utilizes a DBASE II database management system, and a microcomuter environment for implementation at the Air-Base Level. The establishment of this system will relieve the current system from being overloaded, and provide greater economy, better control of stocks by the Air-Base Commander, more efficient part utilization, and more efficient "on time" orders.

APPENDIX A

```
********************** Entry.prg ************************
* AUTHOR    : KONDYLOPOULOS HARALABOS
* DATE      : 9-3-85
* PURPOSE   : THIS PROGRAM PROTECTS THE SYSTEM FROM
*           : UNAUTHORIZED ACCESS
*********************************************************
SET ESCAPE OFF
STORE t TO flag
* Create a continue loop
DO WHILE t
  ERASE
  SET DEFAULT TO b:
  SET COLOR TO 30,14
  @ 1,30 SAY DATE()
  STORE '      ' TO mpassword
  @ 4,10 SAY "ENTER TO THE DBASE, BY WRITING THE CORRECT
  @ 4,12 SAY "PASSWORD"
  @ 12,3 SAY 'ENTER A BLANK TO EXIT '
  @ 23,3 SAY ' ENTER THE PASSWORD '
  SET CONSOLE OFF
  ACCEPT TO mpassword
  SET CONSOLE ON
  READ
    IF $(mpassword,1,1) = ' '
       CANCEL
    ENDIF $(mpassword,1,1)=' '
    IF mpassword =  "GREEK"  .OR. mpassword =  "GOODS"
       ERASE
       SET TALK OFF
       SET COLOR TO 20,10
       @ 5,27 SAY  'WELCOME TO THE SYSTEM'
```

```
      SET COLOR TO 30,14
      @ 8,10 SAY ' THIS SYSTEM ALLOWS THE USER TO IMPROVE
      @ 10,10 SAY ' THE PERFORMANCE OF THE STORAGE/UTILIZA-
      @ 12,10 SAY ' TION OF THE AIRPLANE PARTS.  IT IS A
      @ 14,10 SAY ' COMPLETELY MENU DRIVEN SYSTEM, AND THE
      @ 1     ,10 SAY  ' USER JUST FOLLOW THE INSTRUCTIONS '
      SET COLOR TO 112,30
      @ 23,20 SAY CHR(7)+ " PRESS ANY KEY TO CONTINUE "
      SET CONSOLE OFF
      WAIT
      SET CONSOLE ON
      ERASE
      SET COLOR TO 112,14
      DO menu
   ELSE
      ERASE
      SET COLOR TO 112,140
      @ 8,15 SAY CHR(7)+"TRY AGAIN PLEASE. THIS IS NOT THE
      @ 8,17 SAY " CORRECT PASSWORD "
      @ 8,67 SAY CHR(002)
      SET TALK OFF
      STORE 1 TO z
      DO WHILE z<10
         @ 8,75 SAY CHR(7)
         STORE z+1 TO z
      ENDDO WHILE z<10
      SET COLOR TO 30,14
   ENDIF
ENDDO WHILE t


********************** Menu.prg ***************************
* AUTHOR    : KONDYLOPOULOS HARALABOS
* DATE      : 9-12-85
* PURPOSE   : THIS PROGRAM SELECTS THE DESIRED OPERATION ON
*           : THE SYSTEM BY USING JSING MENU DRIVEN CHOICE
```

```
*************************************************************
SET ESCAPE OFF
SELECT PRIMARY
SET COLOR  TO 30,10
SET FORMAT TO SCREEN
SET CONSOLE ON
SET PRINT OFF
* Create a continue loop
DO WHILE t
        CLEAR
        ERASE
        @ 1,15 SAY " ***' AIRPLANE' PART SYSTEM '***"
        @ 3,1 SAY 'MAIN MENU'
        @ 3,60 SAY DATE()
        @ 7,1 SAY ' OPTIONS:'
        @ 9,1 SAY '     0 EXIT TO OPERATING SYSTEM'
        @ 11,1 SAY '      1 ADDITION OPERATIONS'
        @ 13,1 SAY '      2 DELETION OPERATIONS'
        @ 15,1 SAY '      3 MODIFICATION OPERATIONS'
        @ 9,45 SAY '      4 REPORT OPERATIONS'
        @ 11,45 SAY '      5 QUERIES  '
        @ 13,45 SAY '      6 EXIT TO DBASE'
        @ 20,1 SAY '  SELECT CHOICE '
        STORE ' ' TO choice
        WAIT TO choice
        DO CASE
        CASE choice = '0'
           CLEAR
           QUIT
        CASE choice = '1'
           DO addmenu
           USE
        CASE choice = '2'
           DO delmenu
           USE
```

```
        CASE choice = '3'
           DO modimenu
           USE
        CASE choice = '4'
           DO repomenu
           USE
        CASE choice = '5'
           DO quemenu
           USE
        CASE choice = '6'
           CANCEL
        OTHERWISE
           STORE 1 TO z
           DO WHILE z<5
             @ 30,5 SAY "INVALID CHOICE.. PLEASE TRY AGAIN"
             @ 31,5 SAY CHR(7)
             STORE z+1 TO z
           ENDDO
        ENDCASE
ENDDO WHILE t


*************** Addmenu.prg *******************************
* AUTHOR   : KONDYLOPOULOS HARALABOS
* DATE     : 10-6-85
* PURPOSE  : THIS PROGRAM SELECTS THE REQUIRED ADDITION
*          : OPERATION OF THE SYSTEM BY USING MENU DRIVEN
*          : CHOICE
*****************************************************
SET ESCAPE OFF
SELECT PRIMARY
SET COLOR TO 20,5
SET TALK OFF
SET FORMAT TO SCREEN
SET CONSOLE ON
* Create a continue loop
```

```
DO WHILE t
        CLEAR
        ERASE
        @ 1,15 SAY "*** 'AIRPLANE' PARTS SYSTEM '***"
        @ 3,1 SAY 'ADDMENU'
        @ 3,60 SAY DATE()
        @ 7,1 SAY ' OPTIONS:'
        @ 9,1 SAY '     0 EXIT TO OPERATING SYSTEM'
        @ 11,1 SAY '      1 ADDITION OF PARTS'
        @ 13,1 SAY '      2 ADDITION OF AIRPL_STORES'
        @ 15,1 SAY '      3 ADDITION OF WEAPON_STORES'
        @ 17,1 SAY '      4 ADDITION OF JUNK_STORES'
        @ 19,1 SAY '      5 ADDITION OF ACESSOR_STORES'
        @ 9,45 SAY '      6 ADDITION OF SUPPLIERS'
        @ 11,45 SAY '      7 ADDITION OF INVOICES'
        @ 13,45 SAY '      8 ADDITION OF ORDERS '
        @ 15,45 SAY '      9 EXIT TO DBASE'
        @ 17,45 SAY '      X EXIT TO MAIN MENU'
        @ 20,1 SAY '  SELECT CHOICE  '
        STORE ' ' TO choice
        WAIT TO choice
        DO CASE
        * Execution of the appropriate program
        CASE choice = '0'
           CLEAR
           QUIT
        CASE choice = '1'
           DO adpart
           USE
        CASE choice = '2'
           DO adarstr
           USE
        CASE choice = '3'
           DO adwstr
           USE
```

```
        CASE choice = '4'
           DO adjstr
           USE
        CASE choice = '5'
           DO adacstr
           USE
        CASE choice = '6'
           DO adsuplr
           USE
        CASE choice = '7'
           DO adusage
           USE
        CASE choice = '8'
           DO adorder
           USE
        CASE choice = '9'
           CANCEL
        CASE choice = 'X' .OR. choice = 'x'
           RETURN
           USE
        OTHERWISE
           STORE 1 TO z
           DO WHILE z<10
           @ 30,5 SAY "INVALID CHOICE.. PLEASE TRY AGAIN"
           @ 31,5 SAY CHR(7)
              STORE z+1 TO z
           ENDDO
        ENDCASE
ENDDO WHILE t

************** Adpart.prg ********************************
* AUTHOR     : KONDYLOPOULOS HARALABOS
* DATE       : 10-10-85
* PURPOSE    : THIS PROGRAM ADDS PARTS INTO INVENTORY BY
*            : AUTOMATIC UPDATING OF THE EFFECTED STORE.
```

```
*************************************************************
SET ESCAPE OFF
SET COLOR TO 30,14
SET TALK OFF
STORE t TO flag
* Create a continue loop
DO WHILE t
 SELECT PRIMARY
 USE parts INDEX parts
      STORE '      ' TO mpno
      ERASE
      @ 1,15 SAY "******** 'PART---ADDITION '**********"
      @ 3,1 SAY  'ADDING TO INVENTORY '
      @ 4,60 SAY DATE()
      @ 15,1 SAY 'ENTER PART NUMBER ' GET mpno
      @ 23,1 SAY 'ENTER A BLANK TO EXIT TO ADDMENU'
      READ
      * Return to the addmenu for a blank entry
      IF $(mpno,1,5)= '      '
            RETURN
      ENDIF $(mpno,1,5)= '      '
      FIND &mpno
      * Can not locate the record
      IF #<>0
            SET COLOR TO 112,140
            @ 21,1 SAY 'THAT PART IS ALREADY ON FILE '
            @ 22,40 SAY CHR(7)
            SET TALK OFF
            STORE 1 TO xx
            DC WHILE xx<35
                STORE xx+1 TO xx
            ENDDO WHILE xx<70
            SET COLOR TO 30,14
            LOOP
      ENDIF #<>0
```

```
   ERASE
   @ 1,15 SAY '********** ' PART--ADDITION '***********'
   @ 3,1 SAY 'ADDING TO INVENTORY '
   @ 4,60 SAY DATE ()
   STORE '                ' TO mdesc
   STORE '     ' TO marplt
   STORE '             ' TO mmanfnm
   STORE '             ' TO marstrna
   STORE '    ' TO mjuna
   STORE '             ' TO mwstrna
   STORE '             ' TO macstrna
   STORE 0 TO mqoh
   STORE 0.00 TO mcost
   ERASE
   @ 6,1 SAY 'PART NUMBER        '   +mpno
   @ 7,1 SAY 'DESCRIPTION        ' GET mdesc
   @ 8,1 SAY 'ARPLANE TYPE       ' GET marplt
   @ 9,1 SAY 'SUPPLIER_NAME      ' GET mmanfnm
   @ 10,1 SAY 'ARPL_STORE_NAME    ' GET marstrna
   @ 11,1 SAY 'JUANK_STORE_NAME  ' GET mjuna
   @ 12,1 SAY 'WEAP_STORE_NAME    ' GET mwstrna
   @ 13,1 SAY 'ACCES_STORE_NAME  ' GET macstrna
   @ 14,1 SAY 'QOH                ' GET mqoh
   @ 15,1 SAY 'COST               ' GET mcost
   @ 23,1 SAY 'ENTER A BLANK DESCRIPTION TO EXIT'
   READ
   * Protect the system from addition into more
  * than one file
IF $(macstrna,1,3)<>'   '.AND.$(mjuna,1,3)<>'    '
  .AND.$(mwstrna,1,1)<>'.AND.$(marstrna,1,3)<> '    '
      SET COLOR TO 112,140
      @ 17,15 SAY ' THIS IS NOT PART OF INVENTORY '
      SET COLOR TO 30,14
      LOOP
   ENDIF
```

```
      IF $(mdesc,1,5)='     '
         LOOP
      ENDIF
      * add the record
      APPEND BLANK
      REPLACE acstrna WITH macstrna,desc WITH mdesc
      REPLACE manfnm WITH mmanfnm, arstrna WITH marstrna
     REPLACE arplt WITH marplt, wstrna WITH mwstrna
      REPLACE juna WITH mjuna
      REPLACE pno WITH mpno,qoh WITH mqoh,cost WITH mcost
  * Allow the system to add the entry to the appropriate
  * file
  DO CASE
    CASE marstrna <> '          '
      SELECT SECONDARY
      USE arstores INDEX arstores
      FIND &p.mpno
      STORE '     ' TO maslfno
      ERASE
      @ 6,1 SAY ' ARSTRNA     :' + p.marstrna
      @ 7,1 SAY ' PART NUMBER :' + p.mpno
      @ 8,1 SAY ' DESCRIPTION :' + p.mdesc
      @ 9,1 SAY ' ARPLANE TYPE:' + p.marplt
      @ 10,1 SAY ' QOH         :' + STR(p.mqoh,5)
      @ 11,1 SAY ' COST        :' + STR(p.mcost,8)
      SET COLOR TO 112,140
      @ 12,1 SAY ' ASLFNO ??   :' GET maslfno
      SET COLOR TO 30,14
      READ
      APPEND BLANK
      REPLACE arstrna WITH p.marstrna, pno WITH p.mpno
      REPLACE aslfno WITH maslfno, arplt WITH p.marplt
      REPLACE qoh WITH p.mqoh, desc WITH p.mdesc
REPLACE cost WITH p.mcost
      CLEAR
```

```
    LOOP
  CASE wstrna <> '            '
    SELECT SECONDARY
    USE wstores INDEX wstores
    FIND &p.mpno
    STORE '     ' TO mwslfno
    ERASE
    @ 5,1 SAY 'WPSTORES          :' + p.mwstrna
    @ 6,1 SAY 'PART NUMBER       :' + p.mpno
    @ 7,1 SAY 'DESCRIPTION       :' + p.mdesc
    @ 8,1 SAY 'ARPLAN_TYPE       :' + p.marplt
    @ 9,1 SAY 'QUANTITY ON HAND :' + STR(p.mqoh,5)
    @ 10,1 SAY 'COST              :' + STR(p.mcost,8)
    SET COLOR TO 120,140
    @ 11,1 SAY 'WSLFNO ??         :' GET mwslfno
    SET COLOR TO 30,14
    READ
    APPEND BLANK
    REPLACE wstrna WITH p.mwstrna, pno WITH p.mpno
    REPLACE arplt WITH p.marplt, wslfno WITH mwslfno
    REPLACE qoh WITH p.mqoh, desc WITH p.mdesc
    REPLACE cost WITH p.mcost
    CLEAR
    LOOP
  CASE mjuna <> '    '
    SELECT SECONDARY
    USE junstores INDEX junstores
    FIND &p.mpno
    STORE '      ' TO mjslfno
    ERASE
    @ 5,1 SAY 'JUNSTORES   :' + p.mjuna
    @ 6,1 SAY 'PART_NUMBER :' + p.mpno
    @ 7,1 SAY 'DESCRIPTION :' + p.mdesc
    @ 8,1 SAY 'QUAN_ON_HAND:' + STR(p.mqoh,5)
    @ 9,1 SAY 'COST        :' + STR(p.mcost,8)
```

```
  SET COLOR TO 112,140
  @ 10,1 SAY 'JUANK_SELF# :' GET mjslfno
  SET COLOR TO 30,14
  READ
  APPEND BLANK
  REPLACE juna WITH p.mjuna, pno WITH p.mpno
  REPLACE jslfno WITH mjslfno, qoh WITH p.mqoh
  REPLACE desc WITH p.mdesc, cost WITH p.mcost
  CLEAR
  LOOP
CASE  macstrna  <> '              '
  SELECT SECONDARY
  USE acstores INDEX acstores
  FIND &p.mpno
  STORE '     ' TO macslfno
  ERASE
  @ 5,1 SAY 'ACCESS_STORES :' + macstrna
  @ 6,1 SAY 'PART_NUMBER   :' + p.mpno
  @ 7,1 SAY 'DESCRIPTION   :' + p.mdesc
  @ 8,1 SAY 'ARPL_TYPE     :' + p.marplt
  @ 9,1 SAY 'QUAN_ON_HAND  :' + STR(p.mqoh,5)
  @ 10,1 SAY 'COST          :' + STR(p.mcost,8)
  SET COLOR TO 112,140
  @ 11,1 SAY 'ACCES_SELF_# ?:' GET macslfno
  SET COLOR TO 30,14
  READ
  APPEND BLANK
  REPLACE acstrna WITH macstrna, pno WITH p.mpno
  REPLACE acslfno WITH macslfno, arplt WITH p.marplt
  REPLACE cost WITH p.mcost, acstrna WITH p.acstrna
  REPLACE desc WITH p.mdesc, qoh WITH p.qoh
  CLEAR
  LOOP
OTHERWISE
  @ 18,10 SAY 'THIS IS NOT PART OF THE INVENTORY '
```

```
      CLEAR
      LOOP
  ENDCASE
ENDDO WHILE t

*************** Adsuplr.prg ****************************
*  AUTHOR   : KONDYLOPOULOS HARAL
*  DATE     : 10-25-85
*  PURPOSE  : THIS PROGRAM IS USED FOR ADDING NEW SUPPLIERS
*           : INTO INVENTORY
************************************************************
SET ESCAPE OFF
SET COLOR TO 30,14
SET TALK OFF
SELECT PRIMARY
USE suppliers INDEX suppliers
* enter suppliers
DO WHILE t
      STORE '            ' TO mmanfnm
      ERASE
      @ 1,1 SAY "*** 'SUPPLIERS--ADDITION '***"
      @ 3,1 SAY  'ADDING TO INVENTORY '
      @ 3,60 SAY DATE()
      @ 15,1 SAY "ENTER SUPPLIER'S_ NAME " GET mmanfnm
      @ 23,1 SAY 'ENTER A BLANK TO EXIT'
      STORE !(mmanfnm) TO mmanfnm
      READ
      IF $(mmanfnm,1,5)= '     '
            RETURN
      ENDIF
      FIND &mmanfnm
      IF #<>0
            @ 22,1 SAY "THIS SUPPLIER'S NAME ALREADY "
           @ 22,2 SAY " ON THE FILE "
            @ 22,60 SAY CHR(7)
```

```
      SET TALK OFF
      STORE 1 TO xx
      DO WHILE xx<20
          STORE xx+1 TO xx
      ENDDO WHILE
      LOOP
ENDIF #<>0
ERASE
@ 1,1 SAY '*** 'SUPPLIER'S--ADDITION '***'
@ 2,1 SAY 'ADDING TO INVENTORY '
@ 2,60 SAY DATE ()
STORE '                ' TO mmanfadr
STORE '     ' TO mzip:co
STORE '      ' TO mcountry
STORE '     ' TO marplt
STORE '   ' TO mdel:time
STORE '           ' TO mproduct
ERASE
@ 4,1 SAY "SUPPLIER'S_NAME  "+ mmanfnm
@ 5,1 SAY "SUPPLIER'S ADDRE " GET mmanfadr
@ 6,1 SAY 'ZIP:CODE         ' GET mzip:co
@ 7,1 SAY 'COUNTRY          ' GET mcountry
@ 8,1 SAY 'ARPL_TYPE_SUPPORT' GET marplt
@ 9,1 SAY 'DELIVERY TIME    ' GET mdel:time
@ 10,1 SAY 'PRODUCT          ' GET mproduct
@ 23,1 SAY 'ENTER A BLANK DESCRIPTION TO EXIT'
READ
IF $(mmanfnm,1,1)=' '
   RETURN
ENDIF
* add the record
APPEND BLANK
REPLACE manfnm WITH mmanfnm, manfadr WITH mmanfadr
REPLACE zip:co WITH mzip:co, country WITH mcountry
REPLACE del:time WITH mdel:time, arplt WITH marplt
```

```
      REPLACE product WITH mproduct
ENDDO WHILE t


*************** Adorder ******************************
* AUTHOR     : KONDYLOPOULOS HARALABOS
* DATE       : 10-15-85
* PURPOSE    : THIS PROGRAM ADDS NEW ORDERS INTO ORDER
*            : FILE
************************************************************
SET ESCAPE OFF
SET COLOR TO 30,14
SET TALK OFF
SELECT PRIMARY
USE orders INDEX orders
* enter orders
DO WHILE t
      STORE '       ' TO mordno
      ERASE
      @ 1,1 SAY "********** 'ORDER--ADDITION '**********"
      @ 3,1 SAY  'ADDING TO INVENTORY '
      @ 3,60 SAY DATE()
      @ 15,1 SAY "ENTER ORDER_ NUMBER " GET mordno
      @ 23,1 SAY 'ENTER A BLANK TO EXIT'
      READ
      IF $(mordno,1,5) = '      '
            RETURN
      ENDIF
      FIND &mordno
      IF #<>0
            @ 22,1 SAY 'THIS ORDER NUMBER ALREADY EXIST '
            @ 22,3 SAY ' ON FILE '
            @ 22,60 SAY CHR(7)
            SET TALK OFF
            STORE 1 TO xx
            DO WHILE xx<20
```

```
          STORE xx+1 TO xx
       ENDDO WHILE
       LOOP
ENDIF #<>0
ERASE
@ 1,1 SAY '********** 'ORDER--ADDITION '***********'
@ 2,1 SAY 'ADDING TO INVENTORY '
@ 2,60 SAY DATE ()
STORE '       ' TO mpno
STORE '             ' TO mdesc
STORE '           ' TO mdate
STORE '           ' TO mordna
STORE '             ' TO mmanfnm
STORE 0 TO mqnt
STORE 0.00 TO mcost
ERASE
@ 4,1 SAY "ORDER_NUMBER         " + mordno
@ 5,1 SAY "DATE                 " GET mdate
@ 6,1 SAY 'PNO                  ' GET mpno
@ 7,1 SAY 'DESCRIPTION          ' GET mdesc
@ 8,1 SAY 'NAME OF THE ORDERER  ' GET mordna
@ 9,1 SAY 'SUPPLIER             ' GET mmanfnm
@ 10,1 SAY 'QUANTITY ORDERED     ' GET mqnt
@ 11,1 SAY 'COST                  ' GET mcost
@ 23,1 SAY 'ENTER A BLANK DESCRIPTION TO EXIT'
READ
IF $(mmanfnm,1,1)=' '
   RETURN
ENDIF
* add the record
APPEND BLANK
REPLACE manfnm WITH mmanfnm, ordno WITH mordno
REPLACE pno WITH mpno, desc WITH mdesc, qnt WITH mqnt
REPLACE cost WITH mcost, ordna WITH mordna,
REPLACE date WITH mdate
```

```
ENDDO WHILE t

*************** Adusage.prg ******************************
* AUTHOR    : KONDYLOPOULOS HARALABOS
* DATE      : 10-9-85
* PURPOSE   : THIS PROGRAM ADDS INVOICES INTO USAGE FILE
*           : BY USING THE PART NUMBER
************************************************************
SET COLOR TO 30,14
SET TALK OFF
SET ESCAPE OFF
SELECT PRIMARY
USE usage  INDEX usage
* enter orders
DO WHILE t
      STORE '       ' TO mpno
      ERASE
      @ 1,1 SAY "******** 'INVOICE--ADDITION '********"
      @ 3,1 SAY  'ADDING TO INVENTORY '
      @ 3,60 SAY DATE()
      @ 15,1 SAY "ENTER PART_NUMBER " GET mpno
      @ 23,1 SAY 'ENTER A BLANK TO EXIT'
      READ
      IF $(mpno,1,1)= ' '
            RETURN
      ENDIF
      FIND &mpno
      IF # <> 0
            @ 22,1 SAY 'THIS PART HAVE BEEN REGISTER TO '
@ 22,3 SAY 'THE FILE'
            @ 22,60 SAY CHR(7)
            SET TALK OFF
            STORE 1 TO xx
            DO WHILE xx<20
                STORE xx+1 TO xx
```

```
        ENDDO WHILE
        LCOP
    ENDIF #<>0
    ERASE
    @ 1,1 SAY '******* 'INVOICE--ADDITION '*********'
    @ 2,1 SAY 'ADDING TO INVENTORY '
    @ 2,60 SAY DATE ()
    STORE '     ' TO minvno
    STORE '           ' TO mdesc
    STORE '          ' TO mdate
    STORE '          ' TO muname
    STORE '        ' TO muaddr
    STORE 0 TO mqnt
    STORE 0.00 TO mcost
    ERASE
    @ 4,1 SAY "PART_NUMBER          " + mpno
    @ 5,1 SAY "DATE                 " GET mdate
    @ 6,1 SAY 'INVOICE NUMBER       ' GET minvno
    @ 7,1 SAY 'DESCRIPTION          ' GET mdesc
    @ 8,1 SAY 'NAME OF THE ORDERER ' GET muname
    @ 9,1 SAY 'SUPPLIER             ' GET muaddr
    @ 10,1 SAY 'QUANTITY RECEIVED    ' GET mqnt
    @ 11,1 SAY 'COST                 ' GET mcost
    @ 23,1 SAY 'ENTER A BLANK DESCRIPTION TO EXIT'
    READ
    IF $(minvno,1,1)=' '
       RETURN
    ENDIF
    * add the record
    APPEND BLANK
    REPLACE invno WITH minvno, uname WITH muname
    REPLACE pno WITH mpno, desc WITH mdesc, qnt WITH mqnt
    REPLACE cost WITH mcost, uaddr WITH muaddr
    REPLACE date WITH mdate
ENDDO WHILE t
```

```
********************* Alarstr.prg ************************
* AUTHOR   : KONDYLOPOULOS  HARALABOS
* DATE     : 10-18-85
* PURPOSE  : THIS PROGRAM ADDS NEW AIRPLANE_STORES INTO
*          : THE SYSTEM
************************************************************
SET ESCAPE OFF
SET COLOR TO 30,14
SELECT PRIMARY
USE arstores INDEX arstores
STORE t TO flag
* enter airplane's stores
DO WHILE flag
      STORE '              ' TO marstrna
      ERASE
      @ 1,1 SAY '**** 'ARPL_STORES---ADDITION '****'
      @ 2,1 SAY  'ADDING TO INVENTORY '
      @ 3,60 SAY DATE()
      @ 15,1 SAY 'ENTER ARPL_STORE_NAME ' GET marstrna
      @ 23,1 SAY 'ENTER A BLANK TO EXIT'
      READ
      IF $(marstrna,1,5)= '      '
            RETURN
      ENDIF $(mpno,1,5)= '      '
      FIND &marstrna
      IF #<>0
            @ 22,1 SAY 'THIS ARPL_STORE ALREADY EXIST ON '
            @ 22,3 SAY ' FILE '
            @ 22,60 SAY CHR(7)
            SET TALK OFF
            STORE 1 TO xx
            DC WHILE xx<20
                STORE xx+1 TO xx
            ENDDO WHILE
```

```
            LOOP
        ENDIF #<>0
        ERASE
        @ 1,1 SAY '******** 'ARPL_STORES--ADDITION '*******'
        @ 2,1 SAY 'ADDING TO INVENTORY '
        @ 2,60 SAY DATE ()
        STORE '     ' TO mpno
        STORE '                ' TO mdesc
        STORE '      ' TO maslfno
        STORE '     ' TO marplt
        STORE 0 TO mqoh
        STORE 0.00 TO mcost
        @ 4,1 SAY 'ARPL_STORE_NAME  '+ marstrna
        @ 5,1 SAY 'PART NUMBER      ' GET mpno
        @ 6,1 SAY 'DESCRIPTION      ' GET mdesc
        @ 7,1 SAY 'ARPL_SELF_NUMBER ' GET maslfno
        @ 8,1 SAY 'ARPLANE TYPE     ' GET marplt
        @ 9,1 SAY 'QNT_ON_HAND      ' GET mqoh
        @ 10,1 SAY 'COST             ' GET mcost
        @ 23,1 SAY 'ENTER A BLANK DESCRIPTION TO EXIT'
        READ
        IF $(mpno,1,1)=' '
           RETURN
        ENDIF
        * add the record
        APPEND BLANK
        REPLACE pno WITH mpno,desc WITH mdesc
        REPLACE arstrna WITH marstrna, arplt WITH marplt
        REPLACE aslfno WITH maslfno,qoh WITH mqoh
        REPLACE cost WITH mcost
ENDDO WHILE t


****************** Adwstr.prg ***********************
* AUTHOR     : KONDYLOPOULOS HARALABOS
* DATE       : 10-19-85
```

```
* PURPOSE    : THIS PROGRAM ADDS NEW WEAPON STORES INTO
*            : INVENTORY
*****************************************************
SET ESCAPE OFF
SET COLOR TO 30,14
SELECT PRIMARY
SET TALK OFF
USE wstores INDEX wstores
STORE t TO flag
* enter weapon's stores
DO WHILE flag
      STORE '                ' TO mwstrna
      ERASE
      @ 1,1 SAY '***** 'WEAPON_STORES---ADDITION '******'
      @ 2,1 SAY  'ADDING TO INVENTORY '
      @ 3,60 SAY DATE()
      @ 15,1 SAY 'ENTER WEAPON_STORE_NAME ' GET mwstrna
      @ 23,1 SAY 'ENTER A BLANK TO EXIT'
      READ
      IF $(mwstrna,1,5)= '     '
            RETURN
      ENDIF $(mwstrna,1,5)= '     '
      FIND &mwstrna
      IF #<>0
            @ 22,1 SAY 'THIS WEAPON_STORE ALREADY EXIST '
            @ 22,3 SAY ' ON FILE '
            @ 22,60 SAY CHR(7)
            SET TALK OFF
            STORE 1 TO xx
            DO WHILE xx<20
                STORE xx+1 TO xx
            ENDDO WHILE
            LOOP
      ENDIF #<>0
      ERASE
```

```
      @ 1,1 SAY '**** 'WEAPON_STORES--ADDITION '*****'
      @ 2,1 SAY 'ADDING TO INVENTORY '
      @ 2,60 SAY DATE ()
      STORE '      ' TO mpno
      STORE '                ' TO mdesc
      STORE '     ' TO mwslfno
      STORE '     ' TO marplt
      STORE 0 TO mqoh
      STORE 0.00 TO mcost
      ERASE
      @ 4,1 SAY 'WEAP_STORE_NAME  '+ mwstrna
      @ 5,1 SAY 'PART NUMBER      ' GET mpno
      @ 6,1 SAY 'DESCRIPTION      ' GET mdesc
      @ 7,1 SAY 'WEAP_SELF_NUMBER ' GET mwslfno
      @ 8,1 SAY 'ARPLANE TYPE     ' GET marplt
      @ 9,1 SAY 'QNT_ON_HAND      ' GET mqoh
      @ 10,1 SAY 'COST             ' GET mcost
      @ 23,1 SAY 'ENTER A BLANK DESCRIPTION TO EXIT'
      READ
      IF $(mpno,1,1)=' '
         RETURN
      ENDIF
      * add the record
      APPEND BLANK
      REPLACE pno WITH mpno,desc WITH mdesc
      REPLACE wstrna WITH mwstrna, arplt WITH marplt
      REPLACE wslfno WITH mwslfno,qoh WITH mqoh
      REPLACE cost WITH mcost
ENDDO WHILE t


****************** Adjstr.prg ***************************
* AUTHOR    : KONDYLOPOULOS HARALABOS
* DATE      : 10-19-85
* PURPOSE   : THIS PROGRAM ADDS NEW JUNK STORES INTO THE
*           : SYSTEM
```

```
*************************************************
SET ESCAPE OFF
SET COLOR TO 30,14
SELECT PRIMARY
SET TALK OFF
USE junstores INDEX junstores
STORE t TO flag
* enter junk's stores
* Create a continue loop
DO WHILE flag
      STORE '     ' TO mjuna
      ERASE
      @ 1,1 SAY '***** 'JUNK_STORES---ADDITION '******'
      @ 2,1 SAY  'ADDING TO INVENTORY '
      @ 3,60 SAY DATE()
      @ 15,1 SAY 'ENTER JUNK_STORE_NAME ' GET mjuna
      @ 23,1 SAY 'ENTER A BLANK TO EXIT'
      READ
      IF $(mjuna,1,1) = ' '
            RETURN
      ENDIF $(mjuna,1,1) = ' '
      FIND &mjuna
      IF #<>0
            @ 22,1 SAY 'THIS JUNK_STORE ALREADY EXIST ON '
            @ 22,3 SAY ' FILR '
            @ 22,60 SAY CHR(7)
            SET TALK OFF
            STORE 1 TO xx
            DO WHILE xx<20
                STORE xx+1 TO xx
            ENDDO WHILE
            LOOP
      ENDIF #<>0
      ERASE
      @ 1,1 SAY '**** 'JUNK_STORES--ADDITION '*****'
```

```
      @ 2,1 SAY 'ADDING TO INVENTORY '
      @ 2,60 SAY DATE ()
      STORE '      ' TO mpno
      STORE '                ' TO mdesc
      STORE '     ' TO mjslfno
      STORE '     ' TO marplt
      STORE 0 TO mqoh
      STORE 0.00 TO mcost
      ERASE
      @ 4,1 SAY 'JUNK_STORE_NAME  '+ mjuna
      @ 5,1 SAY 'PART NUMBER      ' GET mpno
      @ 6,1 SAY 'DESCRIPTION      ' GET mdesc
      @ 7,1 SAY 'JUNK_SELF_NUMBER ' GET mjslfno
      @ 8,1 SAY 'QNT_ON_HAND      ' GET mqoh
      @ 9,1 SAY 'COST             ' GET mcost
      @ 23,1 SAY 'ENTER A BLANK DESCRIPTION TO EXIT'
      READ
      IF $(mpno,1,1)=' '
         RETURN
      ENDIF
      * add the record
      APPEND BLANK
      REPLACE pno WITH mpno,desc WITH mdesc,juna WITH mjuna
      REPLACE jslfno WITH mjslfno,qoh WITH mqoh
      REPLACE cost WITH mcost
ENDDO WHILE t


**************** Adacstr.prg ****************************
* AUTHOR     : KONDYLOPOULOS  HARALABOS
* DATE       : 10-2-85
* PURPOSE    : THIS PROGRAM ADDS NEW ACESSORIES STORES INTO
*            : THE SYSTEM
*********************************************************
SET ESCAPE OFF
SET COLOR TO 30,14
```

```
SELECT PRIMARY
USE acstores INDEX acstores
STORE t TO flag
* enter accessories stores
DO WHILE flag
      STORE '            ' TO macstrna
      ERASE
      @ 1,1 SAY '**** 'ACESSORIES_STORES---ADDITION '*****'
      @ 2,1 SAY  'ADDING TO INVENTORY '
      @ 3,60 SAY DATE()
      @ 15,1 SAY 'ENTER ACESSSOR_STORE_NAME ' GET macstrna
      @ 23,1 SAY 'ENTER A BLANK TO EXIT'
      READ
      IF $(macstrna,1,1)= ' '
            RETURN
      ENDIF $(macstrna,1,1)= ' '
      FIND &macstrna
      IF #<>0
            @ 22,1 SAY 'THIS ACESSOR_STORE ALREADY EXIST '
            @ 22,3 SAY ' ON FILE '
            @ 22,60 SAY CHR(7)
            SET TALK OFF
            STORE 1 TO xx
            DC WHILE xx<20
                STORE xx+1 TO xx
            ENDDO WHILE
            LCOP
      ENDIF #<>0
      ERASE
      @ 1,1 SAY '**** 'ACESSORIES_STORES--ADDITION '******'
      @ 2,1 SAY 'ADDING TO INVENTORY '
      @ 2,60 SAY DATE ()
      STORE '      ' TO mpno
      STORE '                ' TO mdesc
      STORE '      ' TO macslfno
```

```
      STORE '      ' TO marplt
      STORE 0 TO mqoh
      STORE 0.00 TO mcost
      @ 4,1 SAY 'ACESS_STORE_NAME  '+ macstrna
      @ 5,1 SAY 'PART NUMBER       ' GET mpno
      @ 6,1 SAY 'DESCRIPTION       ' GET mdesc
      @ 7,1 SAY 'ACESS_SELF_NUMBER' GET macslfno
      @ 8,1 SAY 'ARPLANE TYPE      ' GET marplt
      @ 9,1 SAY 'QNT_ON_HAND       ' GET mqoh
      @ 10,1 SAY 'COST              ' GET mcost
      @ 23,1 SAY 'ENTER A BLANK DESCRIPTION TO EXIT'
      READ
      IF $(mpno,1,1)=' '
         RETURN
      ENDIF
      * add the record
      APPEND BLANK
      REPLACE pno WITH mpno,desc WITH mdesc
      REPLACE acstrna WITH macstrna, arplt WITH marplt
      REPLACE acslfno WITH macslfno, qoh WITH mqon
      REPLACE cost WITH mcost
ENDDO WHILE t


*************** Delmenu.prg ***************************
* AUTHOR     : KONDYLOPCULOS HARALABOS
* DATE       : 10-11-85
* PURPOSE    : THIS PROGRAM SELECTS THE REQUIRED DELETION
*            : OPERATION,BY USING MENU DRIVEN CHOICE
*************************************************
SET ESCAPE OFF
SELECT PRIMARY
SET COLOR TO 20,5
SET TALK OFF
SET FORMAT TO SCREEN
SET CONSOLE ON
```

```
DO WHILE t
        CLEAR
        ERASE
        @ 1,15 SAY "**** 'AIRPLANE' PARTS SYSTEM '****"
        @ 3,1 SAY 'DELMENU'
        @ 3,60 SAY DATE()
        @ 7,1 SAY ' OPTIONS:'
        @ 9,1 SAY '     0 EXIT TO OPERATING SYSTEM'
        @ 11,1 SAY '     1 DELETION OF PARTS'
        @ 13,1 SAY '     2 DELETION OF AIRPL_STORES'
        @ 15,1 SAY '     3 DELETION OF WEAPON_STORES'
        @ 17,1 SAY '     4 DELETION OF SUPPLIERS'
        @ 9,45 SAY '     5 DELETION OF JUNK STORES'
        @ 11,45 SAY '     6 DELETION OF ACCES_STORES'
        @ 13,45 SAY '     7 DELETION OF ORDERS'
        @ 15,45 SAY '     8 DELETION OF INVOICES'
        @ 17,45 SAY '     9 EXIT TO MAIN MENU'
        @ 19,30 SAY '     X EXIT TO DATABASE'
        @ 20,1 SAY '  SELECT CHOICE  '
        STORE ' ' TO choice
        WAIT TO choice
        DO CASE
        CASE choice = '0'
           CLEAR
           QUIT
        CASE choice = '1'
           DO depart
           USE
        CASE choice = '2'
           DO dearstr
           USE
        CASE choice = '3'
           DO dewstr
           USE
        CASE choice = '4'
```

```
            DO desuplr
            USE
         CASE choice = '5'
            DO dejstr
            USE
         CASE choice = '6'
            DO deacstr
            USE
         CASE choice = '7'
            CLEAR
            DO deorder
            USE
         CASE choice = '8'
            DO deusage
            USE
         CASE choice = '9'
            RETURN
            USE
         CASE choice = 'X' .OR. choice = 'x'
            CANCEL
         OTHERWISE
            STORE 1 TO z
            DO WHILE z<5
            @ 30,5 SAY "INVALID CHOICE.. PLEASE TRY AGAIN "
            @ 31,5 SAY CHR(7)
               STORE z+1 TO z
            ENDDO
         ENDCASE
ENDDO WHILE t


********** Depart.prg ************************************
* AUTHOR   : KONDYLOPOULOS HARALABOS
* DATE     : 10-15-85
* PURPOSE  : THIS PROGRAM DELETES PARTS FROM THE INVENTORY
*          : SYSTEM MASTER FILE WITH AUTOMATIC UPDATING
```

```
*          : OF THE EFFECTED FILE
*************************************************************
SET COLOR TO 30,22
DO WHILE t
        SELECT PRIMARY
        USE parts INDEX parts
        ERASE
        STORE '       ' TO mpno
        @ 2,1 SAY 'DELETING FROM PART FILE '
        @ 2,60 SAY DATE()
        @ 4,20 SAY ' ENTER PART NUMBER ' GET mpno
        @ 23,1 SAY ' ENTER A BLANK TO EXIT TO DELMENU '
        READ
        IF $(mpno,1,1) = ' '
             RETURN
        ENDIF
        FIND &mpno
        IF # = 0
           ERASE
           SET COLOR TO 112,140
           @ 22,1 SAY ' THAT PART IS NOT ON THE FILE !!! '
           @ 22,3 SAY ' PLEASE TRY AGAIN '
           SET COLOR TO 30,14
           @ 23,1 SAY CHR(7)
           SET TALK OFF
           STORE 1 TO z
           DO WHILE z<30
              STORE z+1 TO z
           ENDDO WHILE
           LOOP
        ENDIF # = 0
        STORE '                ' TO mdesc
        STORE '     ' TO marplt
        STORE '           ' TO mmanfnm
        STORE '           ' TO marstrna
```

```
     STORE '              ' TO mwstrna
     STORE '              ' TO macstrna
     STORE '    ' TO mjuna
     STORE 0 TO mqoh
     STORE 0.00 TO mcost
     STORE desc TO mdesc
     STORE arplt TO marplt
     STORE manfnm TO mmanfnm
     STORE arstrna TO marstrna
     STORE wstrna TO mwstrna
     STORE juna TO mjuna
     STORE acstrna TO macstrna
     STORE qoh TO mqoh
     STORE cost TO mcost
     ERASE
     @ 2,1 SAY 'DATA OF THE RECORD '
     @ 2,60 SAY DATE()
     @ 6,1 SAY 'PART NUMBER      :'  +mpno
     @ 7,1 SAY 'DESCRIPTION      :'  +mdesc
     @ 8,1 SAY 'AIRPLANE TYPE    :'  +marplt
     @ 9,1 SAY 'SUPPLIERS NAME   :'  +mmanfnm
     @ 10,1 SAY 'ARPL_STORE_NAME  :'  +marstrna
     @ 11,1 SAY 'JUNK_STORE_NAME  :'  +mjuna
     @ 12,1 SAY 'WEAPON_STO_NAME  :'  +mwstrna
     @ 13,1 SAY 'ACCES_STO_NAME   :'  +macstrna
     @ 14,1 SAY 'QOH              :'  +STR(mqoh,5)
     @ 15,1 SAY 'COST             :'  +STR(mcost,8)
     @ 18,20 SAY 'PLEASE BE PATIENT....'
     @ 19,20 SAY 'THIS WILL TAKE TIME !'
     DELETE
     PACK
DO CASE
     CASE marstrna <> '              '
       SELECT SECONDARY
       USE arstores INDEX arstores
```

```
        FIND &mpno
        DELETE
        PACK
        CLEAR
        LOOP
      CASE mwstrna <> '                '
        SELECT SECONDARY
        USE wstores INDEX wstores
        FIND &mpno
        DELETE
        PACK
        CLEAR
        LOOP
      CASE mjuna <> '     '
        SELECT SECONDARY
        USE junstores INDEX junstores
        FIND &mpno
        DELETE
        PACK
        CLEAR
        LOOP
      CASE macstrna <> '               '
        SELECT SECONDARY
        USE acstores INDEX acstores
        FIND &mpno
        DELETE
        PACK
        CLEAR
  ENDDO WHILE t

******************** Desuplr.prg ************************
* AUTHOR   : KONDYLOPOULOS HARALABOS
* DATE     : 10-11-85
* PURPOSE  : THIS PROGRAM DELETES SUPPLIERS OF THE SYSTEM
*************************************************
```

```
SET COLOR TO 30,22
SET TALK OFF
SET ESCAPE OFF
DO WHILE t
        SELECT PRIMARY
        USE suppliers INDEX suppliers
        ERASE
        STORE '            ' TC mmanfnm
        @ 2,1 SAY 'DELETING FROM SUPPLIERS FILE '
        @ 2,60 SAY DATE()
        @ 4,20 SAY ' ENTER SUPPLIER NAME ' GET mmanfnm
        @ 23,1 SAY ' ENTER A BLANK TO EXIT TO DELMENU '
        READ
        IF $(mmanfnm,1,1) = ' '
             RETURN
        ENDIF
        FIND &mmanfnm
        IF # = 0
           ERASE
           SET COLOR TO 112,140
           @ 22,1 SAY ' THAT PART IS NOT ON THE FILE !!! '
           @ 22,3 SAY ' PLEASE TRY AGAIN '
           SET COLOR TO 30,14
           @ 23,1 SAY CHR(7)
           SET TALK OFF
           STORE 1 TO z
           DO WHILE z<30
              STORE z+1 TO z
           ENDDO WHILE
           LOOP
        ENDIF # = 0
        STORE '                ' TO mmanfadr
        STORE '     ' TO mzip:co
        STORE '     ' TO mcountry
        STORE '     ' TO marplt
```

```
        STORE '   ' TO mdel:time
        STORE '           ' TO mproduct
        STORE manfadr TO mmanfadr
        STORE manfnm TO mmanfnm
        STORE zip:co TO mzip:co
        STORE country TO mcountry
        STORE product TO mproduct
        STORE del:time TO mdel:time
        STORE arplt TO marplt
        ERASE
        @ 2,1 SAY 'DATA OF THE RECORD '
        @ 2,60 SAY DATE()
        @ 6,1 SAY 'SUPPLIER NAME     :'
        SET COLOR TO 20,10
        @ 6,20 SAY + !(mmanfnm)
        SET COLOR TO 30,14
        @ 7,1 SAY 'SUPPLIER ADDRESS :'  +mmanfadr
        @ 8,1 SAY 'ZIP CODE         :'  +zip:co
        @ 9,1 SAY 'COUNTRY          :'  +mcountry
        @ 10,1 SAY 'ARPLANE TYPE     :'  +marplt
        @ 11,1 SAY 'DELIVERY TIME    :'  +mdel:time
        @ 12,1 SAY 'PRODUCT          :'  +mproduct
        @ 18,20 SAY 'PLEASE BE PATIENT....'
        DELETE
        PACK
ENDDO WHILE


******************** Deorder.prg **********************
* AUTHOR   : KONDYLOPOULOS  HARALABOS
* DATE     : 10-14-85
* PURPOSE  : THIS PROGRAM DELETE ORDERS FROM THE SYSTEM
*******************************************************
SET COLOR TO 30,22
SET TALK OFF
SET ESCAPE OFF
```

```
DO WHILE t
        SELECT PRIMARY
        USE orders INDEX orders
        ERASE
        STORE '       ' TO mordno
        @ 2,1 SAY 'DELETING FROM ORDER FILE '
        @ 2,60 SAY DATE()
        @ 4,20 SAY ' ENTER ORDER NUMBER ' GET mordno
        @ 23,1 SAY ' ENTER A BLANK TO EXIT TO DELMENU '
        READ
        IF $(mordno,1,1) = ' '
             RETURN
        ENDIF
        FIND &mordno
        IF # = 0
           ERASE
           SET COLOR TO 112,140
           @ 22,1 SAY ' THAT PART IS NOT ON THE FILE !!! '
           @ 22,3 SAY ' PLEASE TRY AGAIN '
           SET COLOR TO 30,14
           @ 23,1 SAY CHR(7)
           SET TALK OFF
           STORE 1 TO z
           DO WHILE z<30
              STORE z+1 TO z
           ENDDO WHILE
           LOOP
        ENDIF # = 0
        STORE '                ' TO mdesc
        STORE '          ' TO mdate
        STORE '            ' TO mmanfnm
        STORE '          ' TO mordna
        STORE '      ' TO mpno
        STORE 0 TO mqnt
        STORE 0.00 TO mcost
```

```
        STORE desc TO mdesc
        STORE date TO mdate
        STORE manfnm TO mmanfnm
        STORE ordna TO mordna
        STORE pno TO mpno
        STORE ordno TO mordno
        STORE qnt TO mqnt
        STORE cost TO mcost
        ERASE
        @ 2,1 SAY 'DATA OF THE RECORD '
        @ 2,60 SAY DATE()
        @ 6,1 SAY 'ORDER NUMBER      :'
        SET COLOR TO 20,10
        @ 6,20 SAY + !(mordno)
        SET COLOR TO 30,14
        @ 7,1 SAY 'DESCRIPTION       :'  +mdesc
        @ 8,1 SAY 'DATE              :'  +mdate
        @ 9,1 SAY 'SUPPLIERS NAME    :'  +mmanfnm
        @ 10,1 SAY 'ORDER NAME        :'  +mordna
        @ 11,1 SAY 'PART NUMBER       :'  +mpno
        @ 12,1 SAY 'QUANTITY          :'  +STR(mqnt,5)
        @ 13,1 SAY 'COST              :'  +STR(mcost,8)
        @ 18,20 SAY 'PLEASE BE PATIENT....'
        DELETE
        PACK
ENDDO WHILE


*************** Deusage.prg ************************
* AUTHOR  : KONDYLOPOULOS  HARALABOS
* DATE    : 10-23-85
* PURPOSE : THIS PROGRAM DELETES INVOICES FROM THE SYSTEM
*************************************************
SET COLOR TO 30,22
SET TALK OFF
SET ESCAPE OFF
```

```
DO WHILE t
        SELECT PRIMARY
        USE usage INDEX usage
        ERASE
        STORE '      ' TO minvno
        @ 2,1 SAY 'DELETING FROM USAGE FILE '
        @ 2,60 SAY DATE()
        @ 4,20 SAY ' ENTER INVOICE NUMBER ' GET minvno
        @ 23,1 SAY ' ENTER A BLANK TO EXIT TO DELMENU '
        READ
        IF $(minvno,1,1) = ' '
             RETURN
        ENDIF
        FIND &minvno
        IF # = 0
           ERASE
           SET COLOR TO 112,140
           @ 22,1 SAY ' THAT PART IS NOT ON THE FILE !!! '
           @ 22,3 SAY ' PLEASE TRY AGAIN '
           SET COLOR TO 30,14
           @ 23,1 SAY CHR(7)
           SET TALK OFF
           STORE 1 TO z
           DO WHILE z<30
              STORE z+1 TO z
           ENDDO WHILE
           LOOP
        ENDIF # = 0
        STORE '                    ' TO mdesc
        STORE '           ' TO mdate
        STORE '           ' TO muname
        STORE '         ' TO muaddr
        STORE '       ' TO mpno
        STORE 0 TO mqnt
        STORE 0.00 TO mcost
```

```
        STORE desc TO mdesc
        STORE date TO mdate
        STORE uname TO muname
        STORE invno TO minvno
        STORE pno TO mpno
        STORE uaddr TO muaddr
        STORE qnt TO mqnt
        STORE cost TO mcost
        ERASE
        @ 2,1 SAY 'DATA OF THE RECORD '
        @ 2,60 SAY DATE()
        @ 6,1 SAY 'INVOICE NUMBER      :'
        SET COLOR TO 20,10
        @ 6,20 SAY + !(minvno)
        SET CCLOR TO 30,14
        @ 7,1 SAY 'DESCRIPTION      :'  +mdesc
        @ 8,1 SAY 'DATE             :'  +mdate
        @ 9,1 SAY 'USER NAME        :'  +muname
        @ 10,1 SAY 'USER ADDRESS     :'  +muaddr
        @ 11,1 SAY 'PART NUMBER      :'  +mpno
        @ 12,1 SAY 'QNT              :'  +STR(mqnt,5)
        @ 13,1 SAY 'COST             :'  +STR(mcost,8)
        @ 18,20 SAY 'PLEASE BE PATIENT....'
        DELETE
        PACK
ENDDO WHILE


***************** Dearstr.prg ****************************
* AUTHOR   : KONDYLOPOULOS  HARALABOS
* DATE     : 10-25-85
* PURPOSE  : THIS PROGRAM DELETES ARPLANE_STORES FROM
*          : THE SYSTEM
**********************************************************
SET COLOR TO 30,22
SET ESCAPE OFF
```

```
SET TALK OFF
ERASE
DO WHILE t
        ERASE
        STORE '              ' TO marstrna
        @ 2,1 SAY 'DELETING FROM AIRPLANE_STORES FILE '
        @ 2,60 SAY DATE()
        @ 4,20 SAY ' ENTER AIRPLSTORE_NAME ' GET marstrna
        @ 23,1 SAY ' ENTER A BLANK TO EXIT TO DELMENU '
        READ
        IF $(marstrna,1,1) = ' '
             RETURN
        ENDIF
        SELECT PRIMARY
        USE arstores INDEX arstores
        FIND &marstrna
        IF # = 0
           SET COLOR TO 112,140
           @ 22,1 SAY ' THAT PART IS NOT ON THE FILE !!! '
           @ 22,3 SAY ' PLEASE TRY AGAIN '
           SET COLOR TO 30,14
           @ 23,1 SAY CHR(7)
           SET TALK OFF
           STORE 1 TO z
           DO WHILE z<30
              STORE z+1 TO z
           ENDDO WHILE
           LOOP
        ENDIF # = 0
        ERASE
        DO WHILE  .NOT. EOF
         IF arstrna = marstrna
            @ 3,1 SAY 'DELETION OF '
            SET COLOR TO 20,10
            @ 3,14 SAY + !(marstrna)
```

```
SET COLOR TO 30,14
@ 4,1 SAY '************'
STORE '                ' TO mdesc
STORE '       ' TO maslfno
STORE '             ' TO marstrna
STORE '       ' TO mpno
STORE 0 TO mqoh
STORE 0.00 TO mcost
ERASE
STORE desc TO mdesc
STORE arplt TO marplt
STORE aslfno TO maslfno
STORE arstrna TO marstrna
STORE pno TO mpno
STORE qoh TO mqoh
STORE cost TO mcost
ERASE
@ 2,1 SAY 'DATA OF THE RECORD '
@ 3,1 SAY 'DELETION OF '
SET COLOR TO 20,10
@ 3,14 SAY + !(marstrna)
SET COLOR TO 30,14
@ 2,60 SAY DATE()
@ 4,1 SAY '************'
@ 6,1 SAY 'ARPL_STORE_NAME  :'  +marstrna
@ 7,1 SAY 'DESCRIPTION      :'  +mdesc
@ 8,1 SAY 'AIRPLANE TYPE    :'  +marplt
@ 9,1 SAY 'SHELF NUMBER     :'  +maslfno
@ 10,1 SAY 'PART NUMBER      :'  +mpno
@ 11,1 SAY 'QOH              :'  +STR(mqoh,5)
@ 12,1 SAY 'COST             :'  +STR(mcost,8)
@ 18,20 SAY 'PLEASE BE PATIENT....'
DELETE
SKIP
LOOP
```

```
          ENDIF
            SKIP
        ENDDO WHILE
            PACK
ENDDO WHILE t


***************** Dewstr.prg ****************************
* AUTHOR     : KONDYLOPOULOS  HARALABOS
* DATE       : 10-28-85
* PURPOSE    : THIS PROGRAM DELETES WEAPON STORES FROM
*            : THE SYSTEM
*************************************************************
SET COLOR TO 30,22
SET ESCAPE OFF
SET TALK OFF
ERASE
DO WHILE t
        ERASE
        STORE '              ' TO mwstrna
        @ 2,1 SAY 'DELETING FROM WEAPON_STORES FILE '
        @ 2,60 SAY DATE()
        @ 4,20 SAY ' ENTER WEAPON_STORE_NAME ' GET mwstrna
        @ 23,1 SAY ' ENTER A BLANK TO EXIT TO DELMENU '
        READ
        IF $(mwstrna,1) = ' '
             RETURN
        ENDIF
        SELECT PRIMARY
        USE wstores INDEX wstores
        FIND &mwstrna
        IF # = 0
           SET COLOR TO 112,140
           @ 22,1 SAY ' THAT PART IS NOT ON THE FILE !!! '
           @ 22,3 SAY ' PLEASE TRY AGAIN '
           SET COLOR TO 30,14
```

```
   @ 23,1 SAY CHR(7)
   SET TALK OFF
   STORE 1 TO z
   DO WHILE z<30
      STORE z+1 TO z
   ENDDO WHILE
   LOCP
ENDIF # = 0
ERASE
DO WHILE  .NOT. EOF
 IF wstrna = mwstrna
    @ 3,1 SAY 'DELETION OF '
    SET COLOR TO 20,10
    @ 3,14 SAY + !(mwstrna)
    SET COLOR TO 30,14
    @ 4,1 SAY '************'
    STORE '                ' TO mdesc
    STORE '       ' TO mslfno
    STORE '             ' TO mwstrna
    STORE '       ' TO mpno
    STORE 0 TO mqoh
    STORE 0.00 TO mcost
    ERASE
    STORE desc TO mdesc
    STORE arplt TO marplt
    STORE wslfno TO mwslfno
    STORE wstrna TO mwstrna
    STORE pno TO mpno
    STORE qoh TO mqoh
    STORE cost TO mcost
    ERASE
    @ 2,1 SAY 'DATA OF THE RECORD '
    @ 3,1 SAY 'DELETION OF '
    SET COLOR TO 20,10
    @ 3,14 SAY + !(mwstrna)
```

```
            SET COLOR TO 30,14
            @ 2,60 SAY DATE()
            @ 4,1 SAY '************'
            @ 6,1 SAY 'ARPL_STORE_NAME  :'  +mwstrna
            @ 7,1 SAY 'DESCRIPTION      :'  +mdesc
            @ 8,1 SAY 'AIRPLANE TYPE    :'  +marplt
            @ 9,1 SAY 'SHELF NUMBER     :'  +mwslfno
            @ 10,1 SAY 'PART NUMBER      :'  +mpno
            @ 11,1 SAY 'QOH              :'  +STR(mqoh,5)
            @ 12,1 SAY 'COST             :'  +STR(mcost,8)
            @ 18,20 SAY 'PLEASE BE PATIENT....'
            DELETE
            SKIP
            LOOP
          ENDIF
            SKIP
       ENDDO WHILE
            PACK
ENDDO WHILE t


***************** Dejstr.prg ***************************
* AUTHOR   : KONDYLOPOULOS HARALABOS
* DATE     : 10-29-85
* PURPOSE  : THIS PROGRAM DELETES JUNK STORES FROM
*          : THE SYSTEM
************************************************************
SET COLOR TO 30,22
SET ESCAPE OFF
SET TALK OFF
ERASE
DO WHILE t
        ERASE
        STORE '    ' TO mjuna
        @ 2,1 SAY 'DELETING FROM JUNK_STORES FILE '
        @ 2,60 SAY DATE()
```

```
@ 4,20 SAY ' ENTER JUNK_STORE_NAME ' GET mjuna
@ 23,1 SAY ' ENTER A BLANK TO EXIT TO DELMENU '
READ
IF $(mjuna,1,1) = ' '
     RETURN
ENDIF
SELECT PRIMARY
USE junstores INDEX junstores
FIND &mjuna
IF # = 0
   SET COLOR TO 112,140
   @ 22,1 SAY ' THAT PART IS NOT ON THE FILE !!! '
   @ 22,3 SAY 'PLEASE TRY AGAIN '
   SET COLOR TO 30,14
   @ 23,1 SAY CHR(7)
   SET TALK OFF
   STORE 1 TO z
   DO WHILE z<30
      STORE z+1 TO z
   ENDDO WHILE
   LOOP
ENDIF # = 0
ERASE
DO WHILE  .NOT. EOF
 IF juna = mjuna
    @ 3,1 SAY 'DELETION OF '
    SET COLOR TO 20,10
    @ 3,14 SAY + !(mjuna)
    SET COLOR TO 30,14
    @ 4,1 SAY '************'
    STORE '                ' TO mdesc
    STORE '       ' TO mjslfno
    STORE '             ' TO mjuna
    STORE '       ' TO mpno
    STORE 0 TO mqoh
```

```
            STORE 0.00 TO mcost
            ERASE
            STORE desc TO mdesc
            STORE jslfno TO mjslfno
            STORE juna TO mjuna
            STORE pno TO mpno
            STORE qoh TO mqoh
            STORE cost TO mcost
            ERASE
            @ 2,1 SAY 'DATA OF THE RECORD '
            @ 3,1 SAY 'DELETION OF '
            SET COLOR TO 20,10
            @ 3,14 SAY + !(mjuna)
            SET COLOR TO 30,14
            @ 2,60 SAY DATE()
            @ 4,1 SAY '************'
            @ 6,1 SAY 'ARPL_STORE_NAME  :'  +mjuna
            @ 7,1 SAY 'DESCRIPTION      :'  +mdesc
            @ 8,1 SAY 'SHELF NUMBER     :'  +mjslfno
            @ 9,1 SAY 'PART NUMBER      :'  +mpno
            @ 10,1 SAY 'QOH              :'  +STR(mqoh,5)
            @ 11,1 SAY 'COST             :'  +STR(mcost,8)
            @ 18,20 SAY 'PLEASE BE PATIENT....'
            DELETE
            SKIP
            LOOP
          ENDIF
            SKIP
        ENDDO WHILE
            PACK
ENDDO WHILE t


**************** Deacstr.prg ******************************
* AUTHOR    : KONDYLOPOULOS HARALABOS
* DATE      : 10-25-85
```

```
* PURPOSE  : THIS PROGRAM DELETES ACESSORIES STORES FROM
*          : THE SYSTEM
*************************************************************
SET COLOR TO 30,22
SET ESCAPE OFF
SET TALK OFF
ERASE
DO WHILE t
        ERASE
        STORE '              ' TO macstrna
        @ 2,1 SAY 'DELETING FROM ACCESS_STORES FILE '
        @ 2,60 SAY DATE()
        @ 4,20 SAY ' ENTER ACCES_STORE_NAME ' GET macstrna
        @ 23,1 SAY ' ENTER A BLANK TO EXIT TO DELMENU '
        READ
        IF $(macstrna,1) = ' '
             RETURN
        ENDIF
        SELECT PRIMARY
        USE acstores INDEX acstores
        FIND &macstrna
        IF # = 0
           SET COLOR TO 112,140
           @ 22,1 SAY ' THAT PART IS NOT ON THE FILE !!! '
           @ 22,3 SAY ' PLEASE TRY AGAIN '
           SET COLOR TO 30,14
           @ 23,1 SAY CHR(7)
           SET TALK OFF
           STORE 1 TO z
           DO WHILE z<30
              STORE z+1 TO z
           ENDDO WHILE
           LOCP
        ENDIF # = 0
        ERASE
```

```
DO WHILE  .NOT. EOF
 IF acstrna = macstrna
    @ 3,1 SAY 'DELETION OF '
    SET COLOR TO 20,10
    @ 3,14 SAY + !(macstrna)
    SET COLOR TO 30,14
    @ 4,1 SAY '************'
    STORE '               ' TO mdesc
    STORE '      ' TO macslfno
    STORE '            ' TO macstrna
    STORE '      ' TO mpno
    STORE 0 TO mqoh
    STORE 0.00 TO mcost
    ERASE
    STORE desc TO mdesc
    STORE arplt TO marplt
    STORE acslfno TO macslfno
    STORE acstrna TO macstrna
    STORE pno TO mpno
    STORE qoh TO mqoh
    STORE cost TO mcost
    ERASE
    @ 2,1 SAY 'DATA OF THE RECORD '
    @ 3,1 SAY 'DELETICN OF '
    SET COLOR TO 20,10
    @ 3,14 SAY + !(macstrna)
    SET COLOR TO 30,14
    @ 2,60 SAY DATE()
    @ 4,1 SAY '************'
    @ 6,1 SAY 'ARPL_STORE_NAME  :'  +nacstrna
    @ 7,1 SAY 'DESCRIPTION      :'  +mdesc
    @ 8,1 SAY 'AIRPLANE TYPE    :'  +marplt
    @ 9,1 SAY 'SHELF NUMBER     :'  +macslfno
    @ 10,1 SAY 'PART NUMBER      :'  +mpno
    @ 11,1 SAY 'QOH              :'  +STR(mqoh,5)
```

```
            @ 12,1 SAY 'COST                :'  +STR(mcost,8)
            @ 18,20 SAY 'PLEASE BE PATIENT....'
            DELETE
            SKIP
            LOOP
          ENDIF
            SKIP
       ENDDO WHILE
            PACK
ENDDO WHILE t


******************** Modimenu.prg ***************************
* AUTHOR  : KONDYLOPOULOS HARALABOS
* DATE    : 10 - 20 - 85
* PURPOSE : THIS PROGRAM ALLOWS THE USER TO SELECT A
*         : CHOICE FOR THE APPROPRIATE FILE MODIFICATION
*************************************************************
SELECT PRIMARY
SET COLOR TO 20,5
SET TALK OFF
SET FORMAT TO SCREEN
SET CONSOLE ON
DO WHILE t
        CLEAR
        ERASE
        @ 1,15 SAY "****** 'AIRPLANE' PARTS SYSTEM '******"
        @ 3,1 SAY 'MODIMENU'
        @ 3,60 SAY DATE()
        @ 7,1 SAY ' OPTIONS:'
        @ 9,1 SAY '     0 EXIT TO OPERATING SYSTEM'
        @ 11,1 SAY '      1 MODIFICATION OF PARTS'
        @ 13,1 SAY '      2 MODIFICATION OF AIRPL_STORES'
        @ 15,1 SAY '      3 MODIFICATION OF WEAPON_STORES'
        @ 17,1 SAY '      4 MODIFICATION OF SUPPLIERS'
        @ 9,45 SAY '      5 MODIFICATTION OF JUNK STORES'
```

```
@ 11,45 SAY '     6 MODIFICATION OF ACCES_STORES'
@ 13,45 SAY '     7 MODIFICATION OF ORDERS '
@ 15,45 SAY '     8 MODIFICATION OF INVOICE '
@ 17,45 SAY '     9 EXIT TO MAIN MENU'
@ 19,26 SAY '    X  EXIT TO DATABASE '
@ 20,1 SAY '  SELECT CHOICE  '
STORE ' ' TO choice
WAIT TO choice
DO CASE
CASE choice = '0'
   CLEAR
   QUIT
CASE choice = '1'
   DO mopart
   USE
CASE choice = '2'
   DO moarstr
   USE
CASE choice = '3'
   DO mowpstr
   USE
CASE choice = '4'
   DO mosuplr
   USE
CASE choice = '5'
   DO mojustr
   USE
CASE choice = '6'
   DO moacstr
   USE
CASE choice = '7'
   DO moorder
   USE
CASE choice = '8'
   DO mousage
```

```
         USE
      CASE choice = '9'
         RETURN
         USE
      CASE choice = 'X' .OR. choice = 'x'
         CANCEL
      OTHERWISE
         STORE 1 TO z
         DO WHILE z<5
         SET COLOR  TO 112,140
         @ 30,5 SAY "INVALID CHOICE... PLEASE TRY AGAIN "
         SET COLOR TO 30,14
         @ 31,5 SAY CHR(7)
            STORE z+1 TO z
         ENDDO
      ENDCASE
ENDDO WHILE t


*************** Mopart.prg ********************************
* AUTHOR  : KONDYLOPOULOS  HARALABOS
* DATE    : 11-2-85
* PURPOSE : THIS PROGRAM MODIFIES PART WITH AUTOMATIC
*         : UPDATING OF THE EFFECTED FILE
*************************************************************
SET ESCAPE OFF
SET COLOR TO 14,30
SELECT PRIMARY
SET TALK OFF
ERASE
*@ 8,18 SAY 'PLEASE BE PATIENT.....'
DO WHILE t
     USE parts INDEX parts
     STORE '      ' TO mpno
     ERASE
     @ 1,20 SAY ' EDITING OF A FILE'
```

```
  @ 3,20 SAY ' ENTER PART_NUMBER' GET mpno
  @ 20,2 SAY 'ENTER A BLANK TO EXIT TO MODIMENU'
  @ 3,50 SAY DATE()
  READ
  IF $(mpno,1,1)=' '
     RETURN
  ENDIF $(mpno,1,1)= ' '
  FIND &mpno
  IF #=0
     ERASE
     @ 13,17 SAY ' THIS RECORD IS NOT IN THE FILE'
     STORE ' ' TO manswer
     ? 'Enter a '
     SET COLOR  TO 120,10
     ??  " ' Y '"
     SET COLOR  TO 30,14
     ?? 'to try again or '
     SET COLOR TO 112,140
     ?? " ' N '"
     SET COLOR TO 30,14
     ?? 'if not and then hit return '
     SET COLOR TO 10,20
  ACCEPT TO manswer
     SET COLOR TO 30,14
     IF manswer  = 'N' .OR. manswer = 'n'
        RETURN
     ELSE
        LOOP
     ENDIF
ENDIF
  ERASE
  @ 1,20 SAY 'EDITING OF A FILE'
  @ 3,50 SAY DATE()
  STORE '                ' TO mdesc
  STORE '     ' TO marplt
```

```
    STORE '              ' TO mmanfnm
    STORE '              ' TO marstrna
    STORE '     ' TO mjuna
    STORE '              ' TO mwstrna
    STORE '              ' TO macstrna
    STORE 0 TO mqoh
    STORE 0.00 TO mcost
    ERASE
    STORE desc TO mdesc
    STORE arplt TO marplt
    STORE manfnm TO mmanfnm
    STORE arstrna TO marstrna
    STORE wstrna TO mwstrna
    STORE juna TO mjuna
    STORE acstrna TO macstrna
    STORE qoh TO mqoh
    STORE cost TO mcost
    * Edit the new values
    @ 8,1 SAY 'PART NUMBER :' +mpno
    @ 9,1 SAY 'DESCRIPTION ' GET mdesc
    @ 10,1 SAY 'ARPLT        ' GET marplt
    @ 11,1 SAY 'MANFNAME     ' GET mmanfnm
    @ 12,1 SAY 'QOH          ' GET mqoh
    @ 13,1 SAY 'COST         ' GET mcost
DO CASE
    CASE marstrna <> '              '
    SET COLOR TO 112,140
    @ 14,1 SAY 'ARPLPA_STO_#' GET marstrna
    CASE mwstrna <> '              '
    SET COLOR TO 20,10
    @ 15,1 SAY 'WEAPON_STO_#' GET mwstrna
    CASE macstrna <> '              '
    SET COLOR TO 112,140
    @ 16,1 SAY 'ACCES_STOR_#' GET macstrna
    CASE mjuna <> '     '
```

```
    SET COLOR TO 20,10
    @ 17,1 SAY 'JUNK_STORE_#' GET mjuna
ENDCASE
    SET COLOR TO 30,14
    READ
    * Replacment of the values
    REPLACE desc WITH mdesc, arplt WITH marplt
    REPLACE wstrna WITH mwstrna, cost WITH mcost
    REPLACE juna WITH mjuna, acstrna WITH macstrna
    REPLACE qoh WITH mqoh, arstrna WITH marstrna
    CASE   arstrna <> '              '
    SELECT SECONDARY
    USE arstores INDEX arstores
    FIND &mpno
    STORE '       ' TO maslfno
    STORE aslfno TO maslfno
    ERASE
    @ 6,1 SAY 'ARPL_STORE_NAME :' + marstrna
    @ 7,1 SAY 'PART_NUMBER     :' + mpno
    @ 8,1 SAY 'DESCRIPTION     :' + mdesc
    @ 9,1 SAY 'ARPL_TYPE       :' + marplt
    @ 10,1 SAY 'QUANT ON HAND   :' + STR(mqoh,5)
    @ 11,1 SAY 'COST            :' + STR(mcost,8)
    SET COLOR TO 112,140
    @ 12,1 SAY 'AIRPL_SHELF_NUMB:' GET maslfno
    SET COLOR TO 30,14
    READ
    REPLACE arstrna WITH marstrna, pno WITH mpno
    REPLACE arplt WITH marplt, cost WITH mcost
    REPLACE  qoh WITH mqoh, aslfno WITH maslfno
    REPLACE desc WITH mdesc
    CLEAR
    LOOP
  CASE wstrna <> '              '
    SELECT SECONDARY
```

```
  USE wstores INDEX wstores
  FIND &mpno
  STORE '      ' TO mwslfno
  STORE wslfno TO mwslfno
  ERASE
  @ 6,1 SAY 'WEAPON_STO_NAME :' + mwstrna
  @ 7,1 SAY 'PART NUMBER     :' + mpno
  @ 8,1 SAY 'DESCRIPTION     :' + mdesc
  @ 9,1 SAY 'ARPLN_TYPE      :' + marplt
  @ 10,1 SAY 'QUANTITY ON HAND:' + STR(mqoh,5)
  @ 11,1 SAY 'COST            :' + STR(mcost,8)
  SET COLOR TO 112,140
  @ 12,1 SAY 'WEAPON_SHELF_#  :' GET mwslfno
  SET COLOR TO 30,14
  READ
  REPLACE wstrna WITH mwstrna, pno WITH mpno
  REPLACE arplt WITH marplt, wslfno WITH mwslfno
  REPLACE cost WITH mcost, cost WITH mcost
  REPLACEdesc WITH mdesc
  CLEAR
  LOOP
CASE juna <> '    '
  SELECT SECONDARY
  USE junstores INDEX junstores
  FIND &mpno
  STORE '      ' TO mjslfno
  STORE jslfno TO mjslfno
  ERASE
  @ 6,1 SAY 'JUNSTORES       :' + mjuna
  @ 7,1 SAY 'PART NUMBER     :' + mpno
  @ 8,1 SAY 'DESCRIPTION     :' + mdesc
  @ 9,1 SAY 'QUAN ON HAND    :' + STR(mqoh,5)
  @ 10,1 SAY 'COST            :' + STR(mcost,8)
  SET COLOR TO 112,140
  @ 11,1 SAY 'JUANK SHELF NUM:' GET mjslfno
```

```
    SET COLOR TO 30,14
    READ
    REPLACE juna WITH mjuna, pno WITH mpno
    REPLACE jslfno WITH mjslfno, qoh WITH mqoh
    REPLACE desc WITH mdesc, cost WITH mcost
    CLEAR
    LOOP
  CASE macstrna <> '            '
    SELECT SECONDARY
    USE acstores INDEX acstores
    FIND &mpno
    STORE '     ' TO macslfno
    STORE acslfno TO macslfno
    ERASE
    @ 6,1 SAY 'ACCESS STORES      :' + macstrna
    @ 7,1 SAY 'PART NUMBER        :' + mpno
    @ 8,1 SAY 'DESCRIPTION        :' + mdesc
    @ 9,1 SAY 'ARPL TYPE          :' + marplt
    @ 10,1 SAY 'QUANT ON HAND      :' + STR(mqoh,5)
    @ 11,1 SAY 'COST               :' + STR(mcost,8)
    SET COLOR TO 112,140
    @ 12,1 SAY 'ACCESS SHELF #     :' GET macslfno
    SET COLOR TO 30,14
    READ
    REPLACE acstrna WITH macstrna, pno WITH mpno
    REPLACE acslfno WITH macslfno, arplt WITH marplt
    REPLACE cost WITH mcost
    REPLACE desc WITH mdesc, qoh WITH mqoh
    CLEAR
    LOOP
  OTHERWISE
    @ 18,10 SAY ' THIS PART IS NOT IN THE INVENTORY !!! '
    CLEAR
    LOOP
ENDCASE
```

```
ENDDO WHILE t

******************** Mosuplr.prg ************************
* AUTHOR   : KONDYLOPOULOS  HARALABOS
* DATE     : 11-2-85
* PURPOSE  : THIS PROGRAM MODIFIES THE SUPPLIER'S DATA
*************************************************************
SET COLOR TO 14,30
SET ESCAPE OFF
SELECT PRIMARY
SET TALK OFF
ERASE
DO WHILE t
     USE suppliers INDEX suppliers
     STORE '           ' TO mmanfnm
     ERASE
     @ 1,20 SAY ' EDITING OF A FILE'
     @ 3,20 SAY ' ENTER SUPPLIER NAME 'GET mmanfnm
     @ 20,2 SAY 'ENTER A BLANK TO EXIT TO MODIMENU'
     @ 3,60 SAY DATE()
     READ
     STORE !(mmanfnm) TO mmanfnm
     IF $(mmanfnm,1,1)=' '
        RETURN
     ENDIF $(mmanfnm,1,1)= ' '
     FIND &mmanfnm
     IF #=0
        ERASE
        @ 13,17 SAY ' THIS RECORD IS NOT IN THE FILE'
        STORE ' ' TO manswer
        ? 'Enter a '
        SET COLOR  TO 120,10
        ??  " ' y '"
        SET COLOR  TO 30,14
        ?? 'to try again or '
```

```
      SET COLOR TO 112,140
      ?? " ' N '"
      SET COLOR TO 30,14
      ?? 'if not and then hit return '
      SET COLOR TO 10,20
   ACCEPT TO manswer
      SET COLOR TO 30,14
      IF manswer  = 'N' .OR. manswer = 'n'
         RETURN
      ELSE
         LOOP
      ENDIF
ENDIF
   ERASE
   @ 1,20 SAY 'EDITING OF A FILE'
   @ 3,50 SAY DATE()
   STORE '                ' TO mmanfadr
   STORE '      ' TO mzip:co
   STORE '       ' TO mcountry
   STORE '      ' TO marplt
   STORE '    ' TO mdel:time
   STORE '            ' TO mproduct
   ERASE
   STORE manfnm TO mmanfnm
   STORE manfadr TO mmanfadr
   STORE zip:co TO mzip:co
   STORE country TO mcountry
   STORE arplt TO marplt
   STORE del:time TO mdel:time
   STORE product TO mproduct
   * Edit the new values
   @ 8,1 SAY 'SUPPLIER NAME:'
   SET COLOR TO 20,10
   @ 8,17 SAY !(+ mmanfnm)
   SET COLOR TO 30,14
```

```
     @ 9,1 SAY 'SUPPLIER ADDRESS' GET mmanfadr
     @ 10,1 SAY 'ZIP:CODE        ' GET mzip:co
     @ 11,1 SAY 'COUNTRY         ' GET mcountry
     @ 12,1 SAY 'ARPLANE TYPE    ' GET marplt
     @ 13,1 SAY 'DELIVERY TIME   ' GET mdel:time
     @ 14,1 SAY 'PRODUCT         ' GET mproduct
     READ
     * Replacment of the values
     REPLACE manfnm WITH mmanfnm, manfadr WITH mmanfadr
     REPLACE zip:co WITH mzip:co, arplt WITH marplt
     REPLACE product WITH mproduct, country WITH mcountry
     REPLACE del:time WITH mdel:time
ENDDO WHILE t


***************** Mousage.prg *************************
* AUTHOR  : KONDYLOPOULOS HARALABOS
* DATE    : 11-3-85
* PURPOSE : THIS PROGRAM MODIFIES THE INVOICE DATA
*******************************************************
SET COLOR TO 14,30
SET ESCAPE OFF
SELECT PRIMARY
SET TALK OFF
ERASE
DO WHILE t
     USE usage INDEX usage
     STORE '     ' TO minvno
     ERASE
     @ 1,20 SAY ' EDITING OF A FILE'
     @ 3,20 SAY ' ENTER INVOICE_NUMBER' GET minvno
     @ 20,2 SAY 'ENTER A BLANK TO EXIT TO MODIMENU'
     @ 3,50 SAY DATE()
     READ
     IF $(minvno,1,1)=' '
        RETURN
```

```
  ENDIF $(minvno,1,1) = ' '
  FIND &minvno
  IF #=0
     ERASE
     @ 13,17 SAY ' THIS RECORD IS NOT IN THE FILE'
     STORE ' ' TO manswer
     ? 'Enter a '
     SET COLOR  TO 120,10
     ??  " ' Y '"
     SET CCLOR  TO 30,14
     ?? 'to try again or '
     SET CCLOR TO 112,140
     ?? " ' N '"
     SET CCLOR TO 30,14
     ?? 'if not and then hit return '
     SET COLOR TO 10,20
  ACCEPT TC manswer
     SET COLOR TO 30,14
     IF manswer  = 'N' .OR. manswer = 'n'
        RETURN
     ELSE
        LOOP
     ENDIF
ENDIF
  ERASE

  @ 1,20 SAY 'EDITING OF A FILE'
  @ 3,50 SAY DATE()
  STORE '                ' TO mdesc
  STORE '      ' TO mpno
  STORE '           ' TO muname
  STORE '           ' TO mdate
  STCRE '         ' TO muaddr
  STORE 0 TO mqnt
  STORE 0.00 TO mcost
  ERASE
```

```
     STORE invno TO minvno
     STORE desc TO mdesc
     STCRE pno TO mpno
     STORE date TO mdate
     STORE uname TO muname
     STORE uaddr TO muaddr
     STORE qnt TO mqnt
     STORE cost TO mcost
     * Edit the new values
     @ 8,1 SAY 'INVOICE NUMBER :'
     SET COLCR TO 20,10
     @ 8,17 SAY !(+ minvno)
     SET COLOR TO 30,14
     @ 9,1 SAY 'DESCRIPTION ' GET mdesc
     @ 10,1 SAY 'DATE        ' GET mdate
     @ 11,1 SAY 'USER NAME   ' GET muname
     @ 12,1 SAY 'USER ADDRESS' GET muaddr
     @ 13,1 SAY 'PART NUMBER ' GET mpno
     @ 14,1 SAY 'QNT         ' GET mqnt
     @ 15,1 SAY 'COST        ' GET mcost
     READ
     * Replacment of the values
     REPLACE desc WITH mdesc, date WITH mdate
     REPLACE pno WITH mpno, invno WITH minvno
     REPLACE qnt WITH mqnt, uaddr WITH muaddr
     REPLACEuname WITH muname, cost WITH mcost
ENDDO WHILE t


********** Moorder.prg **********
* AUTHOR  : KONDYLOPOULOS  HARALABOS
* DATE    : 11-3-85
* PURPOSE : THIS PROGRAM MODIFIES THE ORDERS OF THE
*         : PARTS ON THE ORDER FILE
**********************************************************
SET COLCR TO 14,30
```

```
SET ESCAPE OFF
SELECT PRIMARY
SET TALK OFF
ERASE
* Create a continue loop
DO WHILE t
     USE orders INDEX orders
     STORE '       ' TO mordno
     ERASE
     @ 1,20 SAY ' EDITING OF A FILE'
     @ 3,20 SAY ' ENTER ORDER NUMBER' GET mordno
     @ 20,2 SAY 'ENTER A BLANK TO EXIT TO MODIMENU'
     @ 3,50 SAY DATE()
     READ
     IF $(mordno,1,1)=' '
        RETURN
     ENDIF $(mordno,1,1)= ' '
     FIND &mordno
     IF #=0
        ERASE
        @ 13,17 SAY ' THIS RECORD IS NOT IN THE FILE'
        STORE ' ' TO manswer
        ? 'Enter a '
        SET COLOR  TO 120,10
        ??  " ' Y '"
        SET COLOR  TO 30,14
        ?? 'to try again or '
        SET COLOR TO 112,140
        ?? " ' N '"
        SET COLOR TO 30,14
        ?? 'if not and then hit return '
        SET COLOR TO 10,20
     ACCEPT TO manswer
        SET COLOR TO 30,14
        IF manswer  = 'N' .OR. manswer = 'n'
```

```
        RETURN
     ELSE
        LOCP
     ENDIF
ENDIF
 ERASE
  @ 1,20 SAY 'EDITING OF A FILE'
  @ 3,55 SAY DATE ()
  STORE '                ' TO mdesc
  STORE '       ' TO mpno
  STORE '             ' TO mmanfnm
  STORE '           ' TO mdate
  STORE '           ' TO mordna
  STORE 0 TO mqnt
  STORE 0.00 TO mcost
  ERASE
  STORE desc TO mdesc
  STORE pno TO mpno
  STORE manfnm TO mmanfnm
  STORE date TO mdate
  STORE ordna TO mordna
  STORE qnt TO mqnt
  STORE cost TO mcost
  @ 8,1 SAY 'ORDER NUMBER :'
  SET COLOR TO 20,10
  @ 8,17 SAY !(+ mordno)
  SET COLOR TO 30,14
  @ 9,1 SAY 'DESCRIPTION  ' GET mdesc
  @ 10,1 SAY 'DATE         ' GET mdate
  @ 11,1 SAY 'MANFACT_NAME ' GET mmanfnm
  @ 12,1 SAY 'ORDER NAME   ' GET mordna
  @ 13,1 SAY 'PART NUMBER  ' GET mpno
  @ 14,1 SAY 'QUAN_ON_HAND ' GET mqnt
  @ 15,1 SAY 'COST         ' GET mcost
  READ
```

```
     * Replacment of the values
     REPLACE desc WITH mdesc, date WITH mdate
     REPLACE pno WITH mpno, ordno WITH mordno
     REPLACE qnt WITH mqnt, ordna WITH mordna
     REPLACE manfnm WITH mmanfnm, cost WITH mcost
ENDDO WHILE t


*************** Repomenu.prg ******************************
* AUTHOR   : KONDYLOPOULOS HARALABOS
* DATE     : 10-6-85
* PURPOSE  : THIS PROGRAM SELECTS THE REQUIRED REPORT
*          : OPERATION OF THE SYSTEM BY USING MENU DRIVEN
*          : CHOICE
*************************************************************
SET ESCAPE OFF
SELECT PRIMARY
SET COLOR TO 20,5
SET TALK OFF
SET FORMAT TO SCREEN
SET CONSOLE ON
DO WHILE t
        CLEAR
        ERASE
        @ 1,15 SAY "**** 'AIRPLANE' PARTS SYSTEM '******"
        @ 3,1 SAY 'REPOMENU'
        @ 3,60 SAY DATE()
        @ 7,1 SAY ' OPTIONS:'
        @ 9,1 SAY '     0 EXIT TO OPERATING SYSTEM'
        @ 11,1 SAY '     1 LISTING OF ALL THE INVENTORY'
        @ 13,1 SAY '     2 LISTING OF ALL THE SUPPLIERS'
        @ 15,1 SAY '     3 LISTING OF ALL THE ORDERS'
        @ 17,1 SAY '     4 LISTING OF ALL THE INVOICES'
        @ 19,1 SAY '     5 LISTING OF ALL THE AR_STR PARTS'
        @ 9,38 SAY '     6 LISTING OF ALL THE JU_STR PARTS'
        @ 11,38 SAY '     7 LISTING OF ALL THE WE_STR PARTS'
```

```
@ 13,38 SAY '     8 LISTING OF ALL THE AC_STR PARTS'
@ 15,38 SAY '     9 EXIT TO DATABASE'
@ 17,38 SAY '     X EXIT TO MAIN MENU'
@ 20,1 SAY '  SELECT CHOICE  '
STORE ' ' TO choice
WAIT TO choice
DO CASE
CASE choice = '0'
   CLEAR
   QUIT
CASE choice = '1'
   DO repart
   USE
CASE choice = '2'
   DO resuplr
   USE
CASE choice = '3'
   DO reorder
   USE
CASE choice = '4'
   DO reusage
   USE
CASE choice = '5'
   DO rearstr
   USE
CASE choice = '6'
   DO rejstr
   USE
CASE choice = '7'
   DO rewstr
   USE
CASE choice = '8'
   DO reacstr
   USE
CASE choice = '9'
```

```
            CANCEL
         CASE choice = 'X' .OR. choice = 'x'
            RETURN
            USE
         OTHERWISE
            STORE 1 TO z
            DO WHILE z<10
            @ 30,5 SAY " INVALID CHOICE... PLEASE TRY AGAIN "
            @ 31,5 SAY CHR(7)
               STORE z+1 TO z
            ENDDO
         ENDCASE
ENDDO WHILE t


*************** Repart.prg ****************************
* AUTHOR   : KONDYLOPOULOS  HARALABOS
* DATE     : 11-12-85
* PURPOSE  : THIS PROGRAM LISTS THE INVENTORY OF THE ALL
*          : PARTS OF THE SYSTEM
***********************************************************
SET PRINT OFF
SET ESCAPE OFF
SET COLOR TO 30,14
SET TALK OFF
SET ALTERNATE TO repart.txt
SET ALTERNATE ON
ERASE
@ 2,18 SAY "********° INVENTORY LISTING '**********"
SET COLOR TO 20,10
@ 2,60 SAY DATE()
SET COLOR TO 30,14
USE PARTS
@ 10,18 SAY 'INDEXING..... PLEASE BE PATIENT'
@ 16,18 SAY 'IF YOU WANT TO CANCEL. PLEASE HIT <ESCAPE> '
STORE t TO flag
```

```
DO WHILE flag
   ERASE
   STORE ' ' TO mquery
   ACCEPT " PRINTED INVENTORY ? Y/N " TO mquery
   ERASE
   IF mquery = 'Y'.OR. mquery = 'y'
      SET PRINT ON
      STORE 1 TO c
      DO WHILE c<5
         SET COLOR TO 20,10
         SET PRINT ON
         @ 12,20 SAY 'SET PRINTER ON PLEASE'
         @ 12,50 SAY CHR(7)
         STORE c+1 TO c
         SET COLOR TO 30,14
      ENDDO WHILE
   ELSE
      SET PRINT OFF
   ENDIF
   STORE t TO xflag
   DO WHILE xflag
      ERASE
      SET COLOR TO 112,140
      @ 11,15 SAY ' IF YOU WANT TO FREEZE THE SCREEN'
      @ 12,15 SAY ' DURING THE LISTING, HIT CTRL    NUM_'
      @ 12,15 SAY ' _LOCK TO CONTINUE, PRESS ANY KEY '
      SET COLOR TO 30,14
      @ 1,5 SAY '******** REPORT MENU ********'
      SET COLOR TO 20,10
      @ 3,15 SAY 'ARPLANE TYPE MENU'
      @ 3,60 SAY DATE ()
      SET COLOR TO 30,14
      @ 4,1 SAY 'OPTIONS '
      @ 5,1 SAY '   0   F-104'
      @ 6,1 SAY '   1   F-1. '
```

```
@ 7,1 SAY '   2   F-102'
@ 8,1 SAY '   3   C-130'
@ 9,1 SAY '   4   F-4  '
@ 5,30 SAY '   5   F-5  '
@ 6,30 SAY '   6   F-16 '
@ 7,30 SAY '   7   C-47 '
@ 8,30 SAY '   8   F-18 '
@ 9,30 SAY '   9   PRINT YOUR CHOICE '
@ 10,15 SAY '  X    EXIT TO REPOMENU '
@ 17,1 SAY 'OPTION:'
STORE ' ' TO choice
WAIT TO choice
ERASE
DO CASE
   CASE choice = '0'
      SET COLOR TO 20,10
      @ 14,30 SAY ' PARTS FOR F-104 '
      SET COLOR TO 30,14
      LIST FOR arplt = 'F-104'
   CASE choice = '1'
      SET COLOR TO 20,10
      @ 14,30 SAY 'PARTS OF F-1'
      SET COLOR TO 30,14
      LIST FOR arplt = 'F-1.'
   CASE choice = '2'
      SET COLOR TO 20,10
      @ 14,30 SAY ' PARTS OF F-102 '
      SET COLOR TO 30,14
      LIST FOR arplt = 'F-102'
   CASE choice = '3'
      SET COLOR TO 20,10
      @ 14,30 SAY 'PARTS OF C-130'
      SET COLOR TO 30,14
      LIST FOR arplt = 'C-130'
   CASE choice = '4'
```

```
            SET COLOR TO 20,10
            @ 14,30 SAY ' PARTS OF F-4 '
            SET COLOR TO 30,14
            LIST FOR arplt = 'F-4'
         CASE choice = '5'
            SET COLOR TO 20,10
            @ 14,30 SAY ' PARTS FOR F-5 '
            SET COLOR TO 30,14
            LIST FOR arplt = 'F-5 '
         CASE choice = '6'
            SET COLOR TO 20,10
            @ 14,30 SAY ' PARTS FOR F-16'
            SET COLOR to 30,14
            LIST FOR arplt = 'F-16'
         CASE choice = '7'
            SET COLOR TO 20,10
            @ 14,30 SAY 'PARTS FOR C-47 '
            SET COLOR TO 30,14
            LIST FOR arplt = 'C-47 '
         CASE choice = '8'
            SET COLOR TO 20,10
            @ 14,30 SAY ' PARTS FOR F-18 '
            SET COLOR TO 30,14
            LIST FOR arplt = 'F-18 '
         CASE choice = '9'
            SET PRINT ON
            LOOP
         CASE choice = 'X' .OR. choice = 'x'
           SET PRINT OFF
           RETURN
      ENDCASE
SET ALTERNATE OFF
STORE ' ' TO manswer
SET COLOR TO 20,10
ACCEPT "ENTER A Y IF YOU WISH ANOTHER LISTING, "
```

```
      SET COLOR TO 30,14
      ERASE
      IF manswer = 'y' .OR. manswer = 'Y'
         SET PRINT OFF
         LOOP
      ELSE
         RETURN
      ENDIF
   ENDDO
ENDDO WHILE


**************** Resuplr.prg ****************************
* AUTHOR    : KONDYLOPOULOS  HARALABOS
* DATE      : 11-1-85
* PURPOSE   : THIS PROGRAM REPORTS THE NAMES OF THE
*           : SUPPLIERS
*********************************************************
SET COLOR TO 20,10
SET TALK OFF
DO WHILE t
ERASE
@ 1,12 SAY '****** SUPPLIERS REPORT *********'
@ 3,15 SAY ' IF YOU WANT TO FREEZE THE SCREEN HIT CTRL S '
@ 5,15 SAY ' IF YOU WANT TO RETURN TO THE REPORT MENU, '
@ 5,16 SAY ' HIT ESC'
USE SUPPLIERS
SET COLOR TO 30,14
STORE ' ' TO mquery
ACCEPT "DO YOU WANT THE REPORT PRINTED ? Y/N " TO mquery
 IF mquery = 'Y' .OR. mquery = 'y'
  SET PRINT ON
  STORE 1 TO c
  DO WHILE c<5
     SET COLOR TO 20,10
     @ 12,24 SAY 'PLEASE... SET PRINTER ON '
```

```
     @ 12,50 SAY CHR(7)
     STORE c+1 TO c
     SET COLOR TO 30,14
   ENDDO WHILE
   REPORT FORM SUPPLI
   SET PRINT OFF
 ELSE
   ERASE
   @ 3,15 SAY ' IF YOU WANT TO FREEZE THE SCREEN, '
   @ 5,15 SAY ' HIT CTRLS IF YOU WANT TO RETURN TO REPORT'
   @ 5,16 SAY ' MENU, HIT ESC '
   REPORT FORM SUPPLI
 ENDIF
 STORE ' ' TO mquery
 SET COLOR TO 20,10
 ACCEPT ' DO YOU WANT TO REPEAT THE REPORT ? Y/N ' TO mquery
 SET COLOR TO 30,14
 IF mquery = 'Y' .OR. mquery = 'y'
 LOOP
 ENDIF
 RETURN
ENDDO WHILE


************** Reusage.prg ******************************
* AUTHOR   : KONDYLOPOULOS  HARALABOS
* DATE     : 11-1-85
* PURPOSE  : THIS PROGRAM REPORTS THE USERS OF THE USAGE
*          : FILE
*****************************************************
SET COLOR TO 20,10
SET TALK OFF
DO WHILE t
ERASE
@ 1,15 SAY '************** USERS REPORT ***************'
@ 3,15 SAY ' IF YOU WANT TO FREEZE THE SCREEN HIT CTRL S '
```

```
@ 5,15 SAY ' IF YOU WANT TO RETURN TO THE REPORT MENU,'
@ 5,16 SAY ' HIT ESC'
USE usage
SET COLOR TO 30,14
STORE ' ' TO mquery
ACCEPT "DO YOU WANT THE REPORT PRINTED ? Y/N " TO mquery
 IF mquery = 'Y' .OR. mquery = 'y'
  SET PRINT ON
  STORE 1 TO c
  DO WHILE c<5
     SET COLOR TO 20,10
     @ 12,24 SAY 'PLEASE... SET PRINTER ON '
     @ 12,50 SAY CHR(7)
     STORE c+1 TO c
     SET COLOR TO 30,14
   ENDDO WHILE
   REPORT FORM usa
   SET PRINT OFF
 ELSE
   ERASE
   @ 3,15 SAY ' TO FREEZE THE SCREEN, HIT CTRL    S '
   @ 5,15 SAY ' TO RETURN TO REPORT MENU, HIT ESC '
   ?
   REPORT FORM usa
 ENDIF
STORE ' ' TO mquery
SET COLOR TO 20,10
ACCEPT 'DO YOU WANT TO REPEAT THE REPORT ? Y/N ' TO mquery
SET COLOR TO 30,14
IF mquery = 'Y' .OR. mquery = 'y'
LOOP
ENDIF
RETURN
ENDDO WHILE
```

```
***************** Reorder.prg ******************************
* AUTHOR   : KONDYLOPOULOS  HARALABOS
* DATE     : 11-1-85
* PURPOSE  : THIS PROGRAM REPORTS THE ORDERS OF THE ORDER
*          : FILE
************************************************************
SET COLOR TO 20,10
SET TALK OFF
DO WHILE t
ERASE
@ 1,15 SAY '************** ORDERS REPORT ***************'
@ 3,15 SAY ' TO FREEZE THE SCREEN HIT CTRL   S '
@ 5,15 SAY ' TO RETURN TO THE REPORT MENU,  HIT ESC '
?
USE ORDERS
SET COLOR TO 30,14
STORE ' ' TO mquery
ACCEPT "DO YOU WANT THE REPORT PRINTED ? Y/N " TO mquery
 IF mquery = 'Y' .OR. mquery = 'y'
  SET PRINT ON
  STORE 1 TO c
  DO WHILE c<5
     SET COLOR TO 20,10
     @ 12,24 SAY 'PLEASE... SET PRINTER ON '
     @ 12,50 SAY CHR(7)
     STORE c+1 TO c
     SET COLOR TO 30,14
   ENDDO WHILE
   REPORT FORM ORD
   SET PRINT OFF
 ELSE
   ERASE
   @ 3,15 SAY ' TO FREEZE THE SCREEN, HIT CTRL    S '
   @ 5,15 SAY ' TO RETURN TO REPORT MENU, HIT ESC '
   ?
```

```
   REPORT FORM ORD
 ENDIF
STORE ' ' TO mquery
SET COLOR TO 20,10
ACCEPT 'DO YOU WANT TO REPEAT THE REPORT ? Y/N ' TO mquery
SET COLOR TO 30,14
IF mquery = 'Y' .OR. mquery = 'y'
LOOP
ENDIF
RETURN
ENDDO WHILE

*************** Rearstr.prg ******************************
* AUTHOR    : KONDYLOPOULOS  HARALABOS
* DATE      : 11-12-85
* PURPOSE   : THIS PROGRAM LISTS THE INVENTORY OF THE
*           : AIRPLANE_STORES
************************************************************
SET PRINT OFF
SET ESCAPE OFF
SET COLOR TO 30,14
SET TALK OFF
SET ALTERNATE TO rearstr.txt
SET ALTERNATE ON
ERASE
@ 2,18 SAY "*********° INVENTORY LISTING '**********"
SET COLOR TO 20,10
@ 2,60 SAY DATE()
SET COLOR TO 30,14
USE arstores
@ 10,18 SAY 'INDEXING..... PLEASE BE PATIENT'
@ 16,18 SAY 'IF YOU WANT TO CANCEL. PLEASE HIT <ESCAPE> '
STORE t TO flag
   DO WHILE flag
      ERASE
```

```
STORE ' ' TO mquery
ACCEPT " PRINTED INVENTORY? Y/N " TO mquery
ERASE
IF mquery = 'Y'.OR. mquery = 'y'
   SET PRINT ON
   STORE 1 TO c
   DO WHILE c<5
      SET COLOR TO 20,10
      SET PRINT ON
      @ 12,20 SAY 'SET PRINTER ON PLEASE'
      @ 12,50 SAY CHR(7)
      STORE c+1 TO c
      SET COLOR TO 30,14
   ENDDO WHILE
ELSE
   SET PRINT OFF
ENDIF
STORE t TO xflag
DO WHILE xflag
   ERASE
   SET COLOR TO 112,140
   @ 11,15 SAY ' TO FREEZE THE SCREEN DURING THE '
   @ 12,15 SAY ' LISTING HIT CTRL NUM_LOCK.  '
   @ 12,16 SAY ' PRESS ANY KEY TO CONTINUE '
   SET COLOR TO 30,14
   @ 1,5 SAY '******** REPORT MENU ********'
   SET COLOR TO 20,10
   @ 3,15 SAY 'ARPLANE TYPE MENU'
   @ 3,60 SAY DATE ()
   SET COLOR TO 30,14
   @ 4,1 SAY 'OPTIONS '
   @ 5,1 SAY '   0   F-104'
   @ 6,1 SAY '   1   F-1. '
   @ 7,1 SAY '   2   F-102'
   @ 8,1 SAY '   3   C-130'
```

```
@ 9,1 SAY '   4   F-4  '
@ 5,30 SAY '   5    F-5  '
@ 6,30 SAY '   6    F-16 '
@ 7,30 SAY '   7    C-47 '
@ 8,30 SAY '   8    F-18 '
@ 9,30 SAY '   9    PRINT YOUR CHOICE '
@ 10,15 SAY '  X     EXIT TO REPOMENU '
@ 17,1 SAY 'OPTION:'
STORE ' ' TO choice
WAIT TO choice
ERASE
DO CASE
   CASE choice = '0'
      SET COLOR TO 20,10
      @ 14,30 SAY ' PARTS FOR F-104 '
      SET COLOR TO 30,14
      LIST FOR arplt = 'F-104'
   CASE choice = '1'
      SET COLOR TO 20,10
      @ 14,30 SAY 'PARTS OF F-1'
      SET COLOR TO 30,14
      LIST FOR arplt = 'F-1.'
   CASE choice = '2'
      SET COLOR TO 20,10
      @ 14,30 SAY ' PARTS OF F-102 '
      SET COLOR TO 30,14
      LIST FOR arplt = 'F-102'
   CASE choice = '3'
      SET COLOR TO 20,10
      @ 14,30 SAY 'PARTS OF C-130'
      SET COLOR TO 30,14
      LIST FOR arplt = 'C-130'
   CASE choice = '4'
      SET COLOR TO 20,10
      @ 14,30 SAY ' PARTS OF F-4 '
```

```
         SET COLOR TO 30,14
         LIST FOR arplt = 'F-4'
      CASE choice = '5'
         SET COLOR TO 20,10
         @ 14,30 SAY ' PARTS FOR F-5 '
         SET COLOR TO 30,14
         LIST FOR arplt = 'F-5 '
      CASE choice = '6'
         SET COLOR TO 20,10
         @ 14,30 SAY ' PARTS FOR F-16'
         SET COLOR to 30,14
         LIST FOR arplt = 'F-16'
         @ 14,30 SAY 'PARTS FOR C-47 '
         SET COLOR TO 30,14
         LIST FOR arplt = 'C-47 '
      CASE choice = '8'
         SET COLOR TO 20,10
         @ 14,30 SAY ' PARTS FOR F-18 '
         SET COLOR TO 30,14
         LIST FOR arplt = 'F-18 '
      CASE choice = '9'
         SET PRINT ON
         LOOP
      CASE choice = 'X' .OR. choice = 'x'
        SET PRINT OFF
        RETURN
    ENDCASE
SET ALTERNATE OFF
STORE ' ' TO manswer
SET COLOR TO 20,10
ACCEPT "ENTER A Y IF YOU WISH ANOTHER LISTING, "
SET COLOR TO 30,14
ERASE
IF manswer = 'y' .OR. manswer = 'Y'
   SET PRINT OFF
```

```
         LOOP
      ELSE
         RETURN
      ENDIF
   ENDDO
ENDDO WHILE


***************** Rewstr.prg ****************************
* AUTHOR    : KONDYLOPOULOS  HARALABOS
* DATE      : 11-14-85
* PURPOSE   : THIS PROGRAM LISTS THE INVENTORY OF THE
*           : WEAPON_STORES
*************************************************************
SET PRINT OFF
SET ESCAPE OFF
SET COLOR TO 30,14
SET TALK OFF
SET ALTERNATE TO rewstr.txt
SET ALTERNATE ON
ERASE
@ 2,18 SAY "*********° INVENTORY LISTING '**********"
SET COLOR TO 20,10
@ 2,60 SAY DATE()
SET COLOR TO 30,14
USE wstores
@ 10,18 SAY 'INDEXING..... PLEASE BE PATIENT'
@ 16,18 SAY 'IF YOU WANT TO CANCEL. PLEASE HIT <ESCAPE> '
STORE t TO flag
   DO WHILE flag
      ERASE
      STORE ' ' TO mquery
      ACCEPT " PRINTED INVENTORY ? Y/N " TO mquery
      ERASE
      IF mquery = 'Y'.OR. mquery = 'y'
         SET PRINT ON
```

```
   STORE 1 TO c
   DO WHILE c<5
      SET COLOR TO 20,10
      SET PRINT ON
      @ 12,20 SAY 'SET PRINTER ON PLEASE'
      @ 12,50 SAY CHR(7)
      STORE c+1 TO c
      SET COLOR TO 30,14
   ENDDO WHILE
ELSE
   SET PRINT OFF
ENDIF
STORE t TO xflag
DO WHILE xflag
   ERASE
   SET COLOR TO 112,140
   @ 11,15 SAY ' IF YOU WANT TO FREEZE THE SCREEN ', '
   @ 12,15 SAY ' DURING THE LISTINGHIT CTRL NUM_LOCK.'
   @ 13,16 SAY ' PRESS ANY KEY TO CONTINUE '
   SET COLOR TO 30,14
   @ 1,5 SAY '******** REPORT MENU ********'
   SET COLOR TO 20,10
   @ 3,15 SAY 'ARPLANE TYPE MENU'
   @ 3,60 SAY DATE ()
   SET COLOR TO 30,14
   @ 4,1 SAY 'OPTIONS '
   @ 5,1 SAY '   0   F-104'
   @ 6,1 SAY '   1   F-1. '
   @ 7,1 SAY '   2   F-102'
   @ 8,1 SAY '   3   C-130'
   @ 9,1 SAY '   4   F-4  '
   @ 5,30 SAY '   5   F-5  '
   @ 6,30 SAY '   6   F-16 '
   @ 7,30 SAY '   7   C-47 '
   @ 8,30 SAY '   8   F-18 '
```

```
@ 9,30 SAY '   9   PRINT YOUR CHOICE '
@ 10,15 SAY '  X    EXIT TO REPOMENU '
@ 17,1 SAY 'OPTION:'
STORE ' ' TO choice
WAIT TO choice
ERASE
DO CASE
   CASE choice = '0'
      SET COLOR TO 20,10
      @ 14,30 SAY ' PARTS FOR F-104 '
      SET COLOR TO 30,14
      LIST FOR arplt = 'F-104'
   CASE choice = '1'
      SET COLOR TO 20,10
      @ 14,30 SAY 'PARTS OF F-1'
      SET COLOR TO 30,14
      LIST FOR arplt = 'F-1.'
   CASE choice = '2'
      SET COLOR TO 20,10
      @ 14,30 SAY ' PARTS OF F-102 '
      SET COLOR TO 30,14
      LIST FOR arplt = 'F-102'
   CASE choice = '3'
      SET COLOR TO 20,10
      @ 14,30 SAY 'PARTS OF C-130'
      SET COLOR TO 30,14
      LIST FOR arplt = 'C-130'
   CASE choice = '4'
      SET COLOR TO 20,10
      @ 14,30 SAY ' PARTS OF F-4 '
      SET COLOR TO 30,14
      LIST FOR arplt = 'F-4'
   CASE choice = '5'
      SET COLOR TO 20,10
      @ 14,30 SAY ' PARTS FOR F-5 '
```

```
            SET COLOR TO 30,14
            LIST FOR arplt = 'F-5 '
         CASE choice = '6'
            SET COLOR TO 20,10
            @ 14,30 SAY ' PARTS FOR F-16'
            SET COLOR to 30,14
            LIST FOR arplt = 'F-16'
         CASE choice = '7'
            SET COLOR TO 20,10
            @ 14,30 SAY 'PARTS FOR C-47 '
            SET COLOR TO 30,14
            LIST FOR arplt = 'C-47 '
         CASE choice = '8'
            SET COLOR TO 20,10
            @ 14,30 SAY ' PARTS FOR F-18 '
            SET COLOR TO 30,14
            LIST FOR arplt = 'F-18 '
         CASE choice = '9'
            SET PRINT ON
            LOOP
         CASE choice = 'X' .OR. choice = 'x'
           SET PRINT OFF
           RETURN
      ENDCASE
SET ALTERNATE OFF
STORE ' ' TO manswer
SET COLOR TO 20,10
ACCEPT "ENTER A Y IF YOU WISH ANCTHER LISTING, "
? "ELSE ENTER N " TO manswer
SET COLOR TO 30,14
ERASE
IF manswer = 'y' .OR. manswer = 'Y'
   SET PRINT OFF
   LOOP
ELSE
```

```
         RETURN
      ENDIF
   ENDDO
ENDDO WHILE


********************** Reacstr.prg ************************
* AUTHOR    : KONDYLOPOULOS  HARALABOS
* DATE      : 11-15-85
* PURPOSE   : THIS PROGRAM LISTS THE INVENTORY OF
*           : THE ACCESSORIES_STORES
************************************************************
SET PRINT OFF
SET ESCAPE OFF
SET COLOR TO 30,14
SET TALK OFF
SET ALTERNATE TO reacstr.txt
SET ALTERNATE ON
ERASE
@ 2,18 SAY "*********° INVENTORY LISTING '**********"
SET COLOR TO 20,10
@ 2,60 SAY DATE()
SET COLOR TO 30,14
USE acstores
@ 10,18 SAY 'INDEXING..... PLEASE BE PATIENT'
@ 16,18 SAY 'IF YOU WANT TO CANCEL. PLEASE HIT <ESCAPE> '
STORE t TO flag
   DO WHILE flag
      ERASE
      STORE ' ' TO mquery
      ACCEPT " PRINTED INVENTORY ? Y/N " TO mquery
      ERASE
      IF mquery = 'Y'.OR. mquery = 'y'
         SET PRINT ON
         STORE 1 TO c
         DO WHILE c<5
```

```
      SET COLOR TO 20,10
      SET PRINT ON
      @ 12,20 SAY 'SET PRINTER ON PLEASE'
      @ 12,50 SAY CHR(7)
      STORE c+1 TO c
      SET COLOR TO 30,14
   ENDDO WHILE
ELSE
   SET PRINT OFF
ENDIF
STORE t TO xflag
DO WHILE xflag
   ERASE
   SET COLOR TO 112,140
   @ 11,15 SAY ' IF YOU WANT TO FREEZE THE SCREEN
   @ 12,15 SAY ' DURING THE LISTING HIT CTR NUM_LOCK.'
   @ 12,16 SAY ' PRESS ANY KEY TO CONTINUE.'
   SET COLOR TO 30,14
   @ 1,5 SAY '******** REPORT MENU ********'
   SET COLOR TO 20,10
   @ 3,15 SAY 'ARPLANE TYPE MENU'
   @ 3,60 SAY DATE ()
   SET COLOR TO 30,14
   @ 4,1 SAY 'OPTIONS '
   @ 5,1 SAY '   0   F-104'
   @ 6,1 SAY '   1   F-1. '
   @ 7,1 SAY '   2   F-102'
   @ 8,1 SAY '   3   C-130'
   @ 9,1 SAY '   4   F-4  '
   @ 5,30 SAY '   5   F-5  '
   @ 6,30 SAY '   6   F-16 '
   @ 7,30 SAY '   7   C-47 '
   @ 8,30 SAY '   8   F-18 '
   @ 9,30 SAY '   9   PRINT YOUR CHOICE '
   @ 10,15 SAY '  X    EXIT TO REPOMENU '
```

```
@ 17,1 SAY 'OPTION:'
STORE ' ' TO choice
WAIT TO choice
ERASE
DO CASE
   CASE choice = '0'
      SET COLOR TO 20,10
      @ 14,30 SAY ' PARTS FOR F-104 '
      SET COLOR TO 30,14
      LIST FOR arplt = 'F-104'
   CASE choice = '1'
      SET COLOR TO 20,10
      @ 14,30 SAY 'PARTS OF F-1'
      SET COLOR TO 30,14
      LIST FOR arplt = 'F-1.'
   CASE choice = '2'
      SET COLOR TO 20,10
      @ 14,30 SAY ' PARTS OF F-102 '
      SET COLOR TO 30,14
      LIST FOR arplt = 'F-102'
   CASE choice = '3'
      SET COLOR TO 20,10
      @ 14,30 SAY 'PARTS OF C-130'
      SET COLOR TO 30,14
      LIST FOR arplt = 'C-130'
   CASE choice = '4'
      SET COLOR TO 20,10
      @ 14,30 SAY ' PARTS OF F-4 '
      SET COLOR TO 30,14
      LIST FOR arplt = 'F-4'
   CASE choice = '5'
      SET COLOR TO 20,10
      @ 14,30 SAY ' PARTS FOR F-5 '
      SET COLOR TO 30,14
      LIST FOR arplt = 'F-5 '
```

```
      CASE choice = '6'
         SET COLOR TO 20,10
         @ 14,30 SAY ' PARTS FOR F-16'
         SET COLOR to 30,14
         LIST FOR arplt = 'F-16'
      CASE choice = '7'
         SET COLOR TO 20,10
         @ 14,30 SAY 'PARTS FOR C-47 '
         SET COLOR TO 30,14
         LIST FOR arplt = 'C-47 '
      CASE choice = '8'
         SET COLOR TO 20,10
         @ 14,30 SAY ' PARTS FOR F-18 '
         SET COLOR TO 30,14
         LIST FOR arplt = 'F-18 '
      CASE choice = '9'
         SET PRINT ON
         LOOP
      CASE choice = 'X' .OR. choice = 'x'
        SET PRINT OFF
        RETURN
    ENDCASE
SET ALTERNATE OFF
STORE ' ' TO manswer
SET COLOR TO 20,10
ACCEPT "ENTER A Y IF YOU WISH ANOTHER LISTING,"
? " ELSE ENTER N " TO manswer
SET COLOR TO 30,14
ERASE
IF manswer = 'y' .OR. manswer = 'Y'
   SET PRINT OFF
   LOOP
ELSE
   RETURN
ENDIF
```

```
   ENDDO
ENDDO WHILE


**************** Quemenu.prg ******************************
* AUTHOR     : KONDYLOPOULOS HARALABOS
* DATE       : 10-11-85
* PURPOSE    : THIS PROGRAM SELECTS THE REQUIRED QUERY
*            : OPERATION, BY USING A MENU DRIVEN CHOICE
***************************************************************
SET ESCAPE OFF
SELECT PRIMARY
SET COLOR TO 20,5
SET TALK OFF
SET FORMAT TO SCREEN
SET CONSOLE ON
DO WHILE t
        CLEAR
        ERASE
        @ 1,15 SAY "****** 'AIRPLANE' PARTS SYSTEM '*******"
        @ 3,1 SAY 'QUEMENU'
        @ 3,60 SAY DATE()
        @ 7,1 SAY ' OPTIONS:'
        @ 9,1 SAY '     0 EXIT TO OPERATING SYSTEM'
        @ 11,1 SAY '      1 LOCATION OF A SPECIFIC PART '
        @ 13,1 SAY '      2 COST OF A SPECIFIC PART IN '
        @ 14,17 SAY '      A GIVEN DATE '
        @ 15,1 SAY '      3 NAME OF ORDERED ON A GIVEN '
        @ 16,17 SAY '    PART_NUMBER AND DATE '
        @ 17,1 SAY '      4 EXIT TO MAIN MENU'
        @ 19,1 SAY '      X EXIT TO DATABASE'
        @ 21,1 SAY '  SELECT CHOICE  '
        STORE ' ' TO choice
        WAIT TO choice
        DO CASE
        CASE choice = '0'
```

```
            CLEAR
            QUIT
         CASE choice = '1'
            DO que1
            USE
         CASE choice = '2'
            DO que2
            USE
         CASE choice = '3'
            DO que3
            USE
         CASE choice = '4'
            RETURN
            USE
         CASE choice = 'X' .OR. choice = 'x'
            CANCEL
         OTHERWISE
            STORE 1 TO z
            DO WHILE z<5
            @ 30,5 SAY "INVALID CHOICE.... PLEASE TRY AGAIN "
            @ 31,5 SAY CHR(7)
               STORE z+1 TO z
            ENDDO
         ENDCASE
ENDDO WHILE t


*************** Que1.prg ********************************
* AUTHOR     : KONDYLOPOULOS HARALABOS
* DATE       : 11-11-85
* PURPOSE    : THIS PROGRAM FINDS A SPECIFIC PART FROM THE
*            : INVENTORY
***************************************************************
SET ESCAPE OFF
SET COLOR TO 30,14
SET TALK OFF
```

```
DO WHILE t
 SELECT PRIMARY
 USE parts INDEX parts
      STORE '       ' TO mpno
      ERASE
      @ 4,60 SAY DATE()
      @ 15,1 SAY 'ENTER PART NUMBER ' GET mpno
      @ 23,1 SAY 'ENTER A BLANK TO EXIT TO QUEMENU'
      READ
      SET COLOR TO 20,10
      @ 1,56 SAY mpno
      SET COLOR TO 30,14
      IF $(mpno,1,1)= ' '
            RETURN
      ENDIF $(mpno,1,1)= ' '
      @ 1,15 SAY 'LOCATION OF THE PART WITH PART_NUMBER :'
      FIND &mpno
      IF # = 0
            SET COLOR TO 112,140
            @ 21,1 SAY 'DOES NOT EXIST IN THE FILE '
            @ 22,40 SAY CHR(7)
            SET TALK OFF
            STORE 1 TO xx
            DO WHILE xx<35
                STORE xx+1 TO xx
            ENDDO WHILE xx<70
            SET COLOR TO 30,14
            LOOP
      ENDIF # = 0
      ERASE
      @ 1,15 SAY 'LOCATION OF THE PART WITH PART_NUMBER :'
      SET COLOR TO 20,10
      @ 1,56 SAY mpno
      SET COLOR TO 30,14
      @ 4,60 SAY DATE ()
```

```
    @ 23,1 SAY 'ENTER A BLANK TO EXIT TO QUEMENU'
DO CASE
  CASE arstrna <> '            '
    SELECT SECONDARY
    STORE '     ' TO aslfno
    USE arstores INDEX arstores
    FIND &p.pno
    @ 7,1 SAY ' DESCRIPTION :' + p.desc
    @ 8,1 SAY ' ARPLANE TYPE:' + arplt
    @ 9,1 SAY ' QOH         :' + STR(p.qoh,5)
    @ 10,1 SAY ' COST        :' + STR(p.cost,8)
    SET COLOR to 10,20
    @ 11,1 SAY ' THE PART STORE IS        : '+ arstrna
    @ 12,1 SAY ' THE PART SHELF NUMBER IS : '+ aslfno
    SET COLCR TO 20,10
    @ 23,1 SAY ' TO FREEZE THE SCREEN.. HIT     NUM_LOCK '
    SET COLCR TO 30,14
    READ
    USE
    CLEAR
  CASE wstrna <> '            '
    SELECT SECONDARY
    STORE '     ' TO wslfno
    USE wstores INDEX wstores
    FIND &p.pno
    @ 6,1 SAY 'DESCRIPTION      :' + p.desc
    @ 7,1 SAY 'ARPLAN_TYPE      :' + arplt
    @ 8,1 SAY 'QUANTITY ON HAND :' + STR(p.qoh,5)
    @ 9,1 SAY 'COST             :' + STR(p.cost,8)
    SET COLCR TO 10,20
    @ 10,1 SAY 'THE STORE OF THE PART IS :'+ wstrna
    @ 11,1 SAY 'THE SHELF OF THE PART IS :'+ wslfno
    ST COLOR TO 20,10
    @ 23,1 SAY 'TO FREZE THE SCREEN.. HIT  NUM_LOCK '
    SET COLCR TO 30,14
```

```
  READ
  CLEAR
  LOOP
CASE juna <> '     '
  SELECT SECONDARY
  STORE '       ' TO jslfno
  USE junstores INDEX junstores
  FIND &p.pno
  @ 6,1 SAY 'DESCRIPTION :' +p.desc
  @ 7,1 SAY 'QUAN_ON_HAND:' + STR(p.qoh,5)
  @ 8,1 SAY 'COST         :' + STR(p.cost,8)
  SET COLOR TO 10,20
  @ 9,1 SAY ' THE STORE OF THE PART IS :'+ juna
  @ 10,1 SAY 'THE SHELF OF THE PART IS :'+ jslfno
  SET COLCR TO 20,10
  @ 23,1 SAY 'TO FREEZE THE SCREEN..HIT  NUM_LOCK '
  SET COLCR TO 30,14
  READ
  CLEAR
  LOOP
CASE  acstrna  <> '              '
  SELECT SECONDARY
  STORE '      ' TO acslfno
  USE acstores INDEX acstores
  FIND &p.pno
  @ 6,1 SAY 'DESCRIPTION    :' + p.desc
  @ 7,1 SAY 'ARPL_TYPE      :' + p.arplt
  @ 8,1 SAY 'QUAN_ON_HAND   :' + STR(p.qoh,5)
  @ 9,1 SAY 'COST           :' + STR(p.cost,8)
  SET COLCR TO 10,20
  @ 10,1 SAY 'THE PART STORE IS :' + acstrna
  @ 11,1 SAY 'THE PART SHELF IS :' + acslfno
  SET COLCR TO 20,10
  @ 23,1 SAY ' TO FREEZE THE SCREEN HIT  NUM_LOCK'
  SET COLOR TO 30,14
```

```
      READ
      CLEAR
      LOOP
  OTHERWISE
      @ 18,10 SAY 'THIS IS NOT PART OF THE INVENTORY '
      CLEAR
      LOOP
  ENDCASE
ENDDO WHILE t


*************** Que2.prg ***********************************
* AUTHOR   : KONDYLOPOULOS HARALABOS
* DATE     : 11-18-85
* PURPOSE  : THIS PROGRAM FINDS THE QUANTITY AND THE COST
*          : OF A SPECIFIC PART IN A SPECIFIC TIME.
************************************************************
SET COLOR TO 30,14
SET ESCAPE OFF
SET TALK OFF
ERASE
SELECT PRIMARY
USE orders INDEX pno
STORE t TO flag
* Create a continue loop
DO WHILE t
     STORE ' ' TO answer
     STORE '      ' TO mpno
     @ 15,1 SAY 'ENTER PART NUMBER ' GET mpno
   * If it is selection of new part for the same date
   IF flag
   STORE '          ' TO mdate
   @ 4,60 SAY DATE()
   @ 17,1 SAY 'ENTER THE DATE' GET mdate PICTURE '99-99-99'
   ENDIF
   READ
```

```
* Return to quemenu for a blank entry
IF $(mpno,1,1)= ' '
   RETURN
ENDIF
ERASE
FIND &mpno
IF # = 0
  DO error
ENDIF
STORE 0 TO count
STORE 0 TO temp
* Look up all the file for a specific part# and date
DO WHILE pno = mpno .AND. .NOT. EOF
      IF date = mdate
         ERASE
         @ 6,1 SAY CHR(7)
         SET COLOR TO 20,10
         @ 7,1 SAY 'QUANTITY      COST         EXTER_COST'
         SET COLOR TO 30,14
         @ 10+count,3 SAY +qnt
         @ 10+count,17 SAY +cost
         STORE cost*qnt TO excost
         @ 10+count,36 SAY +excost
         STORE excost+temp TO temp
         STORE count+1 TO count
      ENDIF
         SKIP
ENDDO WHILE
@ 15,15 SAY 'THE TOTAL COST OF THIS DATE, IS :'
SET COLOR TO 20,10
@ 15,49 SAY +temp
STORE 1 TO xx
DO WHILE xx<50
   STORE xx+1 TO xx
ENDDO
```

```
   ERASE
   SET COLOR TO 30,14
   DO WHILE .NOT. ( (answer) = 'N' .OR. (answer) = 'C')
      STORE 'C' TO answer
      SET COLOR TO 112,140
      @ 22,10 SAY ' ENTER AN ACTION                          '
      @ 22,57 GET answer
      @ 23,10 SAY 'C = CONTINUE WITH NEXT PART_NUMBER '
      @ 23,11 SAY ' N TO EXIT TO QUEMENU '
      SET COLOR TO 30,14
      READ
      IF .NOT. ( (answer) = 'N' .OR. (answer) = 'C' )
         DO error
         STORE ' ' TO answer
      ENDIF
   ENDDO WHILE
   DO CASE
   CASE (answer) = 'C'
   * Ask a new part number in the same date
   STORE f TO flag
   LOOP
   CASE (answer) = 'N'
   * Ask a new date for the same part_number
   RETURN
  ENDCASE
ENDDO WHILE


*************** Que3.prg ********************************
* AUTHOR   : KONDYLOPOULOS HARALABOS
* DATE     : 11-18-85
* PURPOSE  : THIS PROGRAM FINDS THE NAME OF THE
*          : PURCHASING AGENT,AND THE TIME OF THE ORDER
*          : FOR A SPECIFIC PART.
***************************************************************
SET COLOR TO 30,14
```

```
SET ESCAPE OFF
SET TALK OFF
ERASE
SELECT PRIMARY
USE orders INDEX pno
STORE t TO flag
* Create a continue loop
DO WHILE t
     STORE ' ' TO answer
     STORE '      ' TO mpno
     @ 15,1 SAY 'ENTER PART NUMBER ' GET mpno
   * If it is selection of new part for the same date
   IF flag
   STORE '          ' TO mdate
   @ 4,60 SAY DATE()
   @ 17,1 SAY 'ENTER THE DATE' GET mdate PICTURE '99-99-99'
   ENDIF
   READ
   * Return to quemenu for a blank entry
   IF $(mpno,1,1)= ' '
      RETURN
   ENDIF
   ERASE
   FIND &mpno
   IF # = 0
     DO error
   ENDIF
   STORE 0 TO count
   STORE 0 TO temp
   * Look up all the file for a specific part# and date
   IF date = mdate .AND. pno = mpno
            ERASE
            @ 6,1 SAY CHR(7)
            SET COLOR TO 20,10
            @ 7,1 SAY 'PA_NUMR - ORDERED_NAME - EXTER_COST'
```

```
          SET COLOR TO 30,14
          @ 10+count,3 SAY +pno
          @ 10+count,17 SAY +ordna
          STORE cost*qnt TO excost
          @ 10+count,36 SAY +excost
ENDIF
@ 15,15 SAY 'THE ORDERED NAME IS :'
SET COLOR TO 20,10
@ 15,37 SAY +ordna
STORE 1 TO xx
DO WHILE xx<50
   STORE xx+1 TO xx
ENDDO
ERASE
SET COLOR TO 30,14
DO WHILE .NOT. ( (answer) = 'N' .OR. (answer) = 'C')
   STORE 'C' TO answer
   SET COLOR TO 112,140
   @ 22,10 SAY ' ENTER AN ACTION                          '
   @ 22,57 GET answer
   @ 23,10 SAY 'C = CONTINUE WITH NEXT PART_NUMBER '
   @ 23,11 SAY ' N = EXIT TO QUEMENU '
   SET COLOR TO 30,14
   READ
   IF .NOT. ( (answer) = 'N' .OR. (answer) = 'C' )
      DO error
      STORE ' ' TO answer
   ENDIF
ENDDO
DO CASE
CASE (answer) = 'C'
* Ask a new part_number in the same date
  STORE f TO flag
  LOOP
CASE (answer) = 'N'
```

```
   * Ask a new order_number for the different date
     RETURN
   ENDCASE
ENDDO WHILE t


************** Error.prg **********************************
* AUTHOR    : KONDYLOPOULOS HARALABOS
* DATE      : 11-11-85
* PURPOSE   : THIS ROUTINE FLASHES A BAD INPUT MESSAGE
*************************************************************
SET COLOR TO errcolor,berrcolor
STORE 1 TO xx
DO WHILE xx<8
  @ 20,50 SAY 'BAD INPUT '
  @ 21,50 SAY 'CHECK YOUR INPUT '
  @ 22,50 SAY 'AND TRY AGAIN '
  @ 22,65 SAY CHR(7)
  STORE xx + 1 TO xx
ENDDO WHILE
ERASE
SET COLOR TO mscolor,ccolor
RETURN
```

APPENDIX B

DEFINITIONS OF THE DATABASE [REF 2]

Binding is the process of fixing data or processing all relationships which are used to generate informations that must be bound. Binding frequently takes place years before a new database system becomes operational.

Computation on a database denotes a section of an application on the database as:

a. Building of the data collection that includes data collection, data organization, and data storage.

b. Updating of a database that includes the addition of new data, the changing of the values, and the deletion of data.

c. Data retrieval that can consist of the fetching of a specific element in order to get a stored value or fact. To fetch a specific data record, the database uses a search argument, which is matched with a key in the database records.

d. Reduction of large quantity of data to usable form we have when the volume summarization of data presented in manageable and usable form. Some of the common used techniques are: statistical summaries and annual business operating statements.

Database is a shared collection of inter-related data designed to meet the varied information needs of an organization.

Database Management System (DBMS) is a software system that performs (carries out) all user's requests for data. User's requests may be an update or a retrieval.

Database system is a system to record and maintain information that is significant to organization in the decision-making processes.

Files is a term in the computer terminology where data is stored more-or-less permanently in a computer and the programs that manipulate it.

Flowcharts is a way of describing complex processes and data flow by using a graphic description. There are three types of activities: control flow, control-data flow, and data flow.

a. Control flow is the sequence of instructions by processing unit.

b. Control-data flow is the flow of data elements that effect the control flow of the computation.

c. Data flow is the typical flow between files and input or output devices.

Input-Output consists of internal and external transfers of data. Internal when reading or writing files where data is transferred between storage unit of the computer and external when reading input or writing output. Database is concerned with data which remains within the scope of the system.

Processes are the basic computational units managed by the operating system. The scheduling and performance of process, and hence of the computations that were specified determine the performance of database systems.

Record is a collection of similar records kept on a secondary computer storage device.

Size is the quantity of data to be handled, and depends on the hardware and operational constraints which may apply in a given environment.

APPENDIX C

DBASE II COMMAND SUMMARY [REF. 17]

"> ? -- displays an expression, variable, or field.
> ?? -- displays an expression list without a preceding line feed.
> @ -- displays user formatted data on screen or printer.
> ACCEPT -- allows input of character strings into memory variables.
> APEND -- append information from another dBASE II database or files in Delimited or System Data format.
> BROWSE -- full screen window viewing and editing of database.
> CANCEL -- cancels command file execution.
> CHANGE -- Non-Full-Screen edit of fields of database.
> CLEAR -- closes databases in use and releases current memory variables.
> CONTINUES -- continue the searching action of a LOCATE command.
> COPY -- creates a copy of an existing database.
> COUNT -- counts the number of records in file which meet some criteria.
> CREATE -- creates new database.
> DELETE -- deletes a file or marks records for deletion.
> DISPLAY -- display files, database records or structure, memory variables, or status.
> DO -- executes command files or structured loops in same.
> EDIT -- initiates edit of records in a database.
> EJECT -- ejects a page on the printer.
> ELSE -- alternate path of command execution within IF.
> ENDCASE -- terminates a CASE command.
> ENDDO -- terminates DO WHILE command.
> ENDIF -- terminates an IF command.
> ENDTEXT -- terminates a TEXT command.
> ERASE -- clears the screen.
> FIND -- positions to record corresponding to key in index file.
> GO or GOTO -- positions to specific record or place in file.
> HELP -- access help file overview or specific help file entry.
> IF -- allows conditional execution of commands.
> INDEX -- creates an index file.
> INPUT -- allows input of expressions into memory variables.
> INSERT -- insert new record in database.

```
> JOIN -- joins output of two databases.
> LIST -- lists files, database records or structure,
          memory variables, and status.
> LOCATE -- find a record that fits a condition.
> LOOP -- skips to beginning of DO WHILE command.
> MODIFY -- create and/or edit command file or modify
            structure of existing database.
> NOTE or * -- allows insertion of comments in command
               file.
> PACK -- erases records marked for deletion.
> QUIT -- exits dBASE and returns to operating system.
> READ -- initiates full-screen editing or formatted
          screen by accepting input into variables
          accessed with a GET.
> RECALL -- erases mark for deletion.
> REINDEX -- update existing index file.
> RELEASE -- eliminates unwanted memory variables and
             releases memory space.
> REMARK -- permits display of any characters.
> RENAME -- rename a file.
> REPLACE -- change information in record(s) or entire
             database field by field.
> REPORT -- format and display a report of information.
> RESET -- reset operating system after placing new disk
           in drive.
> RESTORE -- retrieves memory variables stored in MEM file.
> RETURN -- ends a command file.
> SAVE -- copies current memory variables to disk file.
> SELECT -- switches between USE files in PRIMARY and
            SECONDARY areas.
> SET -- sets dBASE control parameters.
> SKIP -- position forwards or backward in database.
> SORT -- write copy of database sorted on one of the
          data fields.
> STORE -- creates memory variables.
> SUM -- compute and display the sum of database
         field(s).
> TEXT -- allows output of block of text from a
          command file.
> TOTAL -- creates summary copy of database
           combining information from specified
           fields of records meeting some
           criteria.
> UPDATE -- allows batch updates of a database.
> USE -- specifies database to USE until next
         USE command is issued.
> WAIT -- suspends command file processing until
          user input received.
```

APPENDIX D

DBASE II FUNCTIONS [REF. 17]

> @ -- @(<cstring1>, <cstring2>) -- AT function yields an integer whose value is the character number in <cstring2> which begins a substring identical to <cstring1>.

> * -- deleted record function evaluates as a logical TRUE if current record has been marked for deletion.

> # -- record number function reports value of integer corresponding to current record number.

> ! -- !(<cstring>) -- uppercase function yields <cstring> in uppercase characters.

> $ -- $(<cstring>, <start>, <length>) -- substring function forms a character string from the specified part of another another string.

> CHR -- CHR(<numeric exp>) -- yields the ASCII character equivalent of the <numeric exp>. e.g., ? CHR (7) rings the bell.

> DATE() -- returns the character string that contains the System Date in format xx/xx/xx.

> EOF -- end-of-file function evaluates as True if an attempt has been made to go past the last record in a database.

> FILE -- FILE(<file>) -- existence function evaluates as a logical False if it does not.

< INT -- INT (<numeric exp>) -- integer function truncates everything to right of decimal to form an integer.

< LEN -- LEN(<cstring>) -- length function returns the number of characters in <cstring>. ? LEN('HELLO')

< RANK -- RANK(<cstring>) -- returns the (ASCII numeric) value of the leftmost character of <cstring>.

< STR -- STR (<numeric exp>, <width> [,<decimals>]) -- string function converts a numeric expression into a character string.

< TEST -- TEST(<exp>) -- used with ? and IF, test function determines if <exp> is valid and parsable. A valid <exp> returns a value # 0; an invalid <exp> a value of 0.

< TRIM -- TRIM (<cstring>) - trim function removes trailing blanks from <c-string>. ? TRIM(FIRST) + ''+ LAST.

< TYPE -- TYPE(<exp>) -- function yields a one-character string that contains a 'C', 'N', 'L', or 'U' if the <exp. is of type character, Numeric, Logical, or Undefined.

< VAL -- VAL (<cstring>) -- value function converts a character string made of numerals into a numeric expression [Ref. 17].

APPENDIX E

DBASE II LIMITATIONS AND CONSTRAINTS [REF. 17]

number fields/record	32 max
number chars/record	1000 max
number records/database	65535 max
number chars/cstring	254 max
accuracy numeric fields	10 digits
largest number	1.8 x 10**63 approx
number current memyars	64 max
number chars/command line	254 max
number <exp>s in SUM command	5 max
number chars in REPORT header	254 max
number fields in REPORT	24 max
number chars in index key	99 max
number of pending GETS	64 max
number of files open at one tim	16 max
length of .PRG file	unlimited

LIST OF REFERENCES

1. Barston, David R., Interactive Programming Environment, p. 17, McGraw Hill, Inc., 1984.

2. MacLennan, Bruce J., Functional Programming Methodology, CS 4150 Course Notes, 1985.

3. Wiederhold, Gio, Database Design, p. 1, McGraw Hill, 1977.

4. Hsiao, David K., Advanced Database Machine Architecture, Prentice Hall, 1983.

5. Senn, James A., Information Systems in Management, Wadsworth Publishers, 1982.

6. Ullman, Jeffrey D., Database Systems, Computer Software Engineering Series, 1982.

7. Townsend, Carl, Using dBASE II, McGraw Hill, 1984.

8. Squires, Stephen L., Branstad, Martha, Zelkowitz, Marvin, Rapid Prototyping Workshop: An Overview; Software Engineering Notes, Volume 7, Number 3, July 1982.

9. Gregory, S. T., "On Prototypes vs Mockups," ACM Sigsoft Engineering Notes, P. 13, Volume 9, Number 5, October 1984.

10. Wetherbe, James C., Systems Analysis and Design, West Publishing Company, 1984.

11. Bonet, Rafael, Kung, Antonio, TECSI, "Software, Structuring Subsystems: The Experience of Prototyping," Approach-ACM Sigsoft Software Engineering, Volume 9, Number 5, October 1984.

12. Sommerville, I., Software Engineering, Addison-Wesley, 1982.

13 Fairley, Richard, Software Engineering, McGraw Hill Inc., 1985.

14 Meyer, Ken, and Northwood, Almos Kovacs, Middlesex, England, "Readers' Forum", Datamation, page 252, September 1983.

15. Sehan, Amrun, System for Microcomputers, Masters Thesis, Naval Postgraduate School, December 1979.

16. Relue, Richard B., Comparison of Microcomputer Based Database Management, Masters Thesis, Naval Postgraduate School, June, 1982.

17. Lyons, Norman, IS4183 Course Notes, 1985.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5100	2
3.	Computer Technology Curricular Office Naval Postgraduate School Code 37 Monterey, California 93943-5100	1
4.	Department Chairman Computer Science Department Code 52 Naval Postgraduate School Monterey, California 93943-5100	1
5.	CDR L.C. Rawlinson, Code 52Rv Naval Postgraduate School Monterey, California 93943-5100	1
6.	LCDR P. W. Callahan, Code 52CS Naval Postgraduate School Monterey, California 93943-5100	1
7.	Major H. P. Kondylopoulos Embassy of Greece Office of Air Attache 2228 Massachusetts Avenue, N.W. Washington, D. C. 20008	3

216700
Thesis
K767 Kondylopoulos
c.1 The use of DBASE II
and microcomputers.